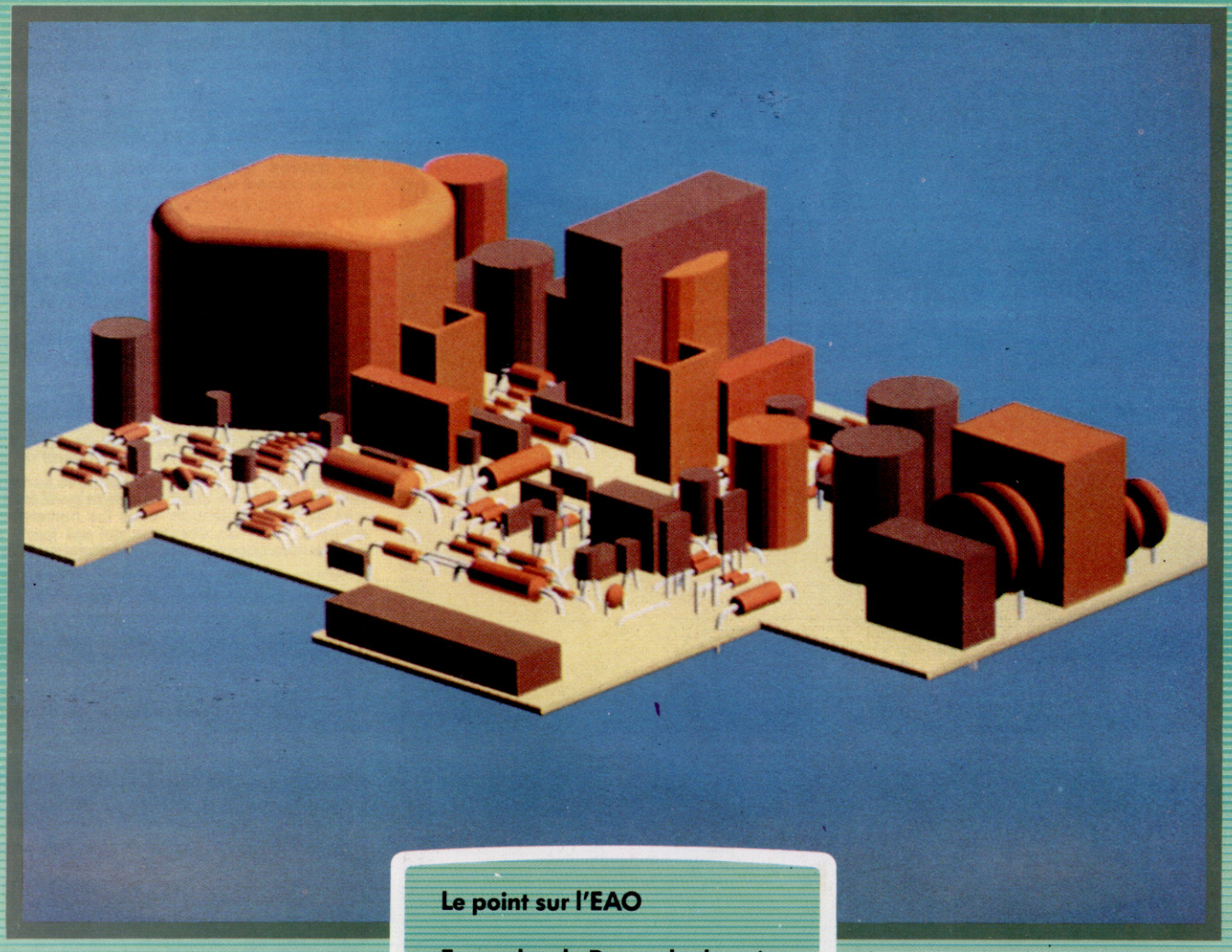
OBC INFORMATIQUE

N° 84

COURS D'INFORMATIQUE PRATIQUE ET FAMILIALE



- Le point sur l'EAO
- Exemples de Bases de données
- Robots en kit
- La puce vidéo du C64

EDITIONS ATLAS

M 6062-84-12,50 F 90 FB - 3,80 FS - $ 1.95

Page manquante
(publicité et colophon)

Formes du futur

Nous terminons notre suite d'articles sur l'enseignement par un tour d'horizon des nouvelles perspectives offertes par l'ordinateur en matière de méthodes pédagogiques.

L'Enseignement assisté par ordinateur (E.A.O.) est en train de connaître une véritable révolution. Matériel moins cher, logiciel plus abouti et améliorations technologiques concourent à changer radicalement l'enseignement auquel nous sommes habitués. Loin d'être le parent pauvre des marchés informatiques, l'enseignement s'impose de plus en plus comme le secteur de pointe du développement d'une nouvelle technologie. Ce renversement de tendance s'explique aisément. L'enseignement fut en effet d'abord négligé par les informaticiens, constructeurs et éditeurs de logiciels, au profit du marché du consommateur, beaucoup plus lucratif... La situation change rapidement avec la prise de conscience de l'importance des marchés éducatifs. L'enfant devient une cible pour les publicitaires, qui voient en lui un consommateur futur.

La conséquence immédiate de cette évolution est l'amélioration du matériel destiné à l'enseignement (moins cher et davantage accessible). Apple lança en 1985 une campagne concernant la vente de Macintosh aux écoles avec un rabais de 30 %. Atari suivit Apple en créant une nouvelle gamme de micros ST de 32 bits destinés à l'enseignement. Son originalité était de proposer

La classe du futur
L'informatique peut révolutionner notre conception de l'enseignement en faisant de l'enfant le centre d'intérêt de la méthode pédagogique. C'est lui-même qui commande l'accès aux diverses ressources éducatives (les informations) mises à sa disposition par les nouvelles technologies. Les transmissions par satellite et les systèmes de réseau permettent aux étudiants de consulter des bases de données dites « intelligentes » à l'autre bout du monde. L'accès à d'autres cultures est ainsi facilité. Les écoles deviendront des « centres supervisant l'acquisition des connaissances ». Les enfants s'y rencontreront pour confronter leurs travaux, dont la quasi-totalité pourra être faite à domicile. Des systèmes vidéo interactifs permettront le développement de logiciels extrêmement sophistiqués. Des robots pourront être utilisés pour des démonstrations de concepts physiques abstraits.
(Cl. Alan Adler.)

LOGO en mémoire morte à la place du BASIC. En France, par exemple, Thomson, Exelvision, Goupil ont été choisis pour s'implanter dans les écoles. Les marchés en jeu sont considérables, pouvant aller jusqu'à 100 machines par jour.

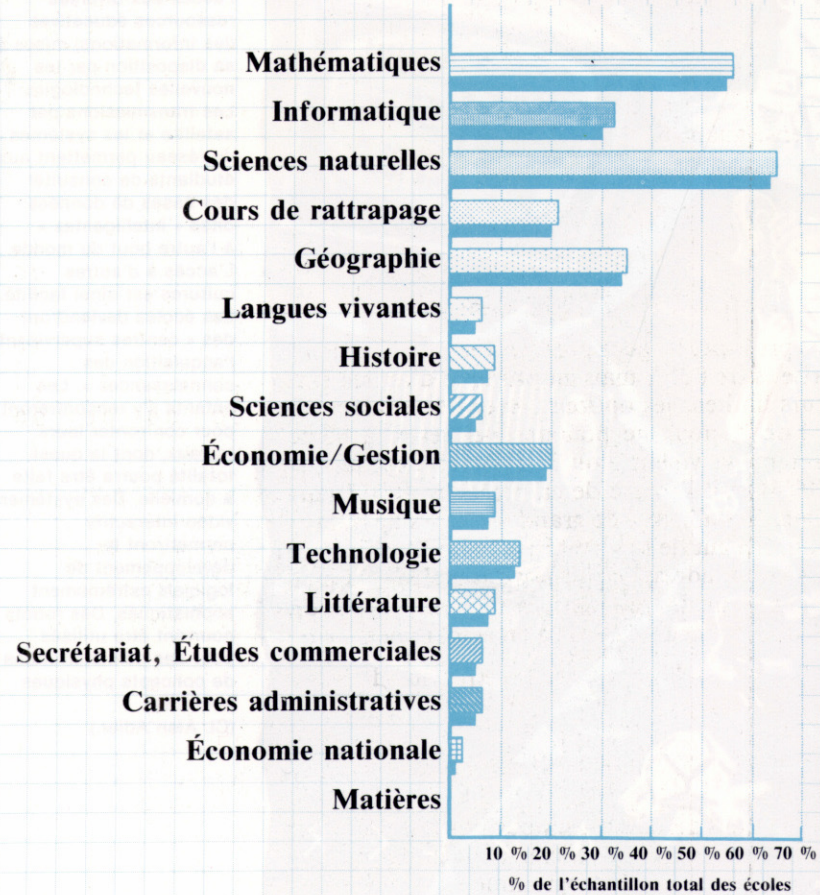
En dehors de l'aspect commercial du marché éducatif, il s'est également fait jour une nouvelle génération d'utilisateurs de micros, qui est plus exigeante et recherche des applications plus sérieuses. Le développement de programmes éducatifs a maintenant débordé le cadre de l'école, et rencontre des marchés plus importants. La création de didacticiels est donc devenue plus rigoureuse. Le manque de matériel et la pauvreté des logiciels ne freinant plus le développement de l'E.A.O., que peut-on attendre dans ce domaine? L'utilisation de l'ordinateur peut maintenant s'orienter dans deux directions : l'extension des techniques existantes et l'apparition de méthodes radicalement nouvelles.

Seymour Papert a déjà indiqué un certain nombre d'améliorations à apporter au LOGO en utilisant la puissance de machines telles que le Macintosh. Papert voudrait voir incorporer au langage les lois élémentaires de la physique. Cette « micro-physique » permettrait aux enfants de jouer avec les lois de la gravité, comme ils le font avec celles de la géométrie. Il prévoit également une base de données « micro-universelle » dans laquelle les enfants pourraient librement puiser des informations sur le traitement des données. Certaines de ces idées ont d'ores et déjà été mises en œuvre dans un logiciel, Mac'LOGO. Les simulations, les programmes d'E.A.O. et les bases de données deviendront plus accessibles et effectifs avec l'entrée à l'école de programmes de consultation tels que Filevision de Macintosh.

Cependant, les perspectives les plus prometteuses en matière d'enseignement par ordinateur tiennent à l'introduction de technologies entièrement nouvelles, telles que les mémoires mortes sur disques compacts ou les vidéos interactives. Une grande capacité de stockage à un prix abordable permet de libérer les didacticiels de leur contrainte d'organisation linéaire, caractéristique des programmes développés sur de petits systèmes non conviviaux. Le degré d'interaction d'un programme d'E.A.O. de 48 K est extrêmement limité, imposant des contraintes à la méthode d'acquisition des connaissances. La vidéo interactive permet une plus grande souplesse dans le traitement du sujet, utilisant l'espace mémoire accru pour autoriser plusieurs cheminements pendant l'exécution des programmes.

On peut distinguer quatre domaines d'application pour l'utilisation de la vidéo interactive en matière d'enseignement par ordinateur. Les professeurs pourraient l'utiliser comme aide pédagogique par le biais de programmes éducatifs et de références audiovisuelles. L'accès à des banques d'images et de films s'en trouverait considérablement accéléré. De petits groupes d'élèves pourraient l'utiliser comme tuteur, étu-

Vraisemblance vidéo
L'utilisation par matière des micro-ordinateurs concerne avant tout les disciplines scientifiques. L'introduction des systèmes vidéo interactifs et le développement de logiciels sur cette même base devraient ouvrir l'informatique aux autres domaines. Par exemple, l'enseignement de l'histoire pourrait être radicalement changé par la pratique de simulations complexes mettant à profit les capacités de stockage gigantesques des disques vidéo.
(Cl. Ian McKinnell.)



Domesday + 900

Neuf cents ans après Guillaume le Conquérant et son cadastre britannique, le vidéodisque et son énorme capacité de stockage de données sont mis à l'épreuve pour réunir toutes sortes de données sur la Grande-Bretagne. La BBC, Philips, Thorn-Emi ainsi que l'organisme d'informatisation des écoles britanniques ont mis en œuvre ce projet gigantesque.
Les écoles doivent collecter des données sur leur environnement immédiat, l'ensemble étant réuni pour former une banque de données publique. Celle-ci comportera des informations sur l'utilisation des terres, les services et associations diverses, etc. En outre, les sources nationales de données devront être fusionnées (par exemple, le recensement ou les études sur les transports) en une seule et immense base de données nationale.
Les deux bases tiendront sur un seul disque laser. En outre, la technologie du disque laser autorise l'enregistrement de cartes, de diagrammes, de tables et même de séquences de dessins animés.

diant la documentation et répondant aux questions. Les utilisateurs individuels s'en serviraient pour des simulations entièrement interactives, des didacticiels complexes et des programmes d'enseignement très ouverts. En dernier lieu, il ferait office d'accès à des données, comme bibliothèque audio et vidéo.

Bien qu'il reste beaucoup de travail à faire en ce qui concerne l'indexation et l'accès aux informations avec vidéo interactive, le système est déjà bien introduit dans de nombreuses applications. Une université britannique l'a utilisé, dans ses cours d'été, pour simuler l'effet de forces physiques sur la matière. Un autre programme l'a mis à profit pour créer des masques, des effets d'animation et des séquences vidéo. Il s'agissait en l'occurrence d'un programme mettant en scène une audience d'un tribunal opposant des fabricants d'ours en peluche (!) et de machines à laver (...), le sujet de controverse étant de savoir si les yeux desdits ours résisteraient au traitement des machines à laver. L'étudiant-interlocuteur de la vidéo interactive doit mener à bien un certain nombre d'expériences ayant valeur d'expertise. Le programme est contrôlé par un didacticiel traitant les causes possibles de défaillance des matériaux intervenant dans la fabrication des yeux. L'étudiant effectue diverses expériences et en interprète les résultats, ces derniers étant finalement mis en scène.

Plus d'écoles

Le transfert des techniques logicielles actuelles — simulation, didacticiels, bases de données, etc. — vers des systèmes à mémoire morte sur disque compact est fructueux ; mais les effets à long terme de cette nouvelle technologie vont beaucoup plus loin. Selon Seymour Papert, « il n'y aura plus d'écoles d'ici vingt ans. L'idée d'un professeur faisant face à une classe prenant des notes est incompatible avec l'informatique. Si l'on passe en revue les aspects actuels de l'école, on s'aperçoit qu'ils deviennent inutiles et inefficaces. Par contre, ce qui est aujourd'hui freiné — la communication entre enfants — sera favorisé... Il y aura alors une forme d'organisation (qu'on l'appelle "école" ou autrement), un lieu où les enfants se rencontreront et développeront ensemble leur personnalité. Ce qui est important, c'est que l'école d'aujourd'hui n'existera plus ».

Son opinion se trouve développée par le professeur Tom Stonier, qui affirme : « L'ordinateur marquera le début du transfert de l'éducation de l'école à la maison. »

Les implications possibles de cette mutation d'une structure pédagogique, centrée sur le système scolaire, à une organisation privilégiant l'individu sont considérables et dépassent le cadre de l'école.

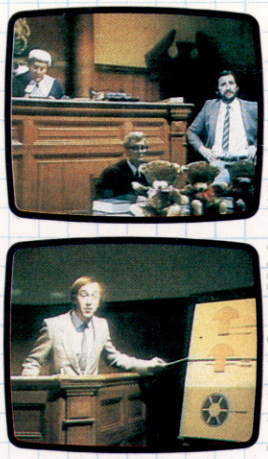
Selon Stonier, avec l'école à la maison et une plus longue espérance de vie (améliorations dues aux progrès de la médecine), les personnes âgées s'occuperont davantage de l'éducation des enfants, comme c'est le cas dans les sociétés dites « primitives ». Stonier estime que cette ressource inexploitée est tout indiquée pour prendre en charge une partie de l'enseignement.

Il existe aux États-Unis une illustration de ce que pourrait être un tel système. A Portage, Wisconsin, un professeur rend visite à chaque enfant une fois par semaine. Il apprécie à cette occasion les progrès faits en une semaine, et décide avec les parents du travail à donner à l'enfant.

Bien que l'exemple de Portage concerne des enfants handicapés, il pourrait tout aussi bien s'appliquer à d'autres étudiants. En Grande-Bretagne, il est légal d'instruire vous-même, à domicile, vos enfants, si pouvez prouver que vous êtes en mesure de le faire. Il y a actuellement plus de deux mille familles ayant ainsi retiré leurs enfants de l'école. L'accès à de grandes bases de données publiques, l'utilisation de micros puissants à mémoire résidant sur disque compact (bases de données) et de logiciels interactifs donneront aux parents-éducateurs de puissants outils d'enseignement à domicile.

Avec l'accroissement en complexité et en diversité des didacticiels, le rôle des professeurs devient de moins en moins aisé à définir. Stonier prédit que les enfants deviendront davantage aptes que leurs professeurs à acquérir des connaissances.

En effet, aussi longtemps que les professeurs font office de « base de données ambulantes », les connaissances acquises par l'enfant sont strictement limitées par l'intelligence même de son professeur. Tous les professeurs d'informatique (et d'autres matières) savent qu'il y a des élèves qui les ont dépassés. Stonier pense qu'il y en aura de plus en plus, avec le développement des bases de données et des didacticiels. Les professeurs deviendront des conseillers, experts en pédagogie, et non spécialistes d'une matière donnée. Ils guideront les élèves dans l'acquisition des connaissances, en les suivant, et non en les « enseignant ».



Production du Centre BBC Open University

Témoin occulaire
Les séquences filmées et autres informations visuelles de haute résolution supposent le stockage et le traitement d'énormes quantités de données. Avec les systèmes contrôlés par ordinateur, c'était tout à fait impossible. L'arrivée de la technologie de la vidéo interactive le permet. Ces photos d'écran provenant du disque de l'« ours en peluche » illustrent un système de disque vidéo contrôlé par ordinateur. Il s'agit ici d'un cours destiné à des élèves ingénieurs.
Le programme met en scène un rapport d'expertise devant un tribunal. Il simule des expériences et établit l'arrêt du tribunal.

ROM sur disque compact

Pour sauvegarder les données de manière fiable, une grande partie de la ROM sur disque est occupée par des informations relatives au dépistage d'erreurs et au formatage. Néanmoins, un seul disque compact offre une capacité de stockage effective de 55 M, ce qui correspond à des centaines de pages-écrans.
En outre, contrairement au disque audio qui suppose des convertisseurs complexes digital/analogique, les lecteurs à ROM sur disque compact n'ont affaire qu'à des données numériques. Cela simplifie la circuiterie et réduit les coûts à l'unité. En sachant que les lecteurs sont par ailleurs pratiquement identiques aux lecteurs audio, les prix sont susceptibles de baisser avec la demande et l'amélioration des techniques de fabrication.
Les disques compacts audio coûtent environ 120 F ; les ROM sur disques compacts devraient donc se vendre à des prix similaires si la production est suffisamment importante. Du fait qu'un seul disque ROM peut certainement contenir tous les logiciels dont vous pouvez avoir besoin, leur introduction sur le marché micro-informatique est sur le point de révolutionner le marché logiciel.

Variables importantes

Nous examinons ici l'utilisation des fonctions et donnons quelques exemples qui illustrent l'intégrité structurale stricte et concise du langage PASCAL.

Comme les procédures qui peuvent être définies et appelées ultérieurement, les fonctions peuvent aussi être appelées, mais uniquement à titre d'expressions — non comme instructions. Alors que des appels de procédures entraînent l'exécution de sous-programmes pour effectuer un traitement quelconque, les appels de fonction ne font que calculer une valeur. Le résultat renvoyé peut être de type simple, réel ou scalaire, ou de type « pointeur » (que nous avons déjà rencontré). La seule différence, en dehors de l'utilisation du mot réservé FUNCTION, est que l'identificateur de type du résultat donné doit être spécifié. Supposons que PASCAL n'ait pas la fonction prédéfinie odd — nous pourrions facilement définir la nôtre :

```
FUNCTION Odd (nombre:entier):booléen;
  BEGIN
     Odd:=nombre MOD2>0
  END; {Odd}
```

La valeur est renvoyée dans l'identificateur de la fonction et, par conséquent, doit lui être attribuée dans le corps de la fonction. De ce point de vue, les noms de fonction sont similaires à des variables qui ne sont jamais initialisées, mais leurs valeurs sont calculées lorsqu'elles apparaissent dans une expression. Elles ne peuvent apparaître que du côté gauche d'une affectation; ainsi, si nous programmons l'affectation de la façon suivante :

```
IF nombre MOD 2>0 THEN
  Odd:=vrai
```

(en oubliant la clause nécessaire ELSE Odd:=faux), il est possible de passer outre à cette construction, laissant ainsi le résultat indéfini.
Comme pour les paramètres de procédures que nous avons examinés jusqu'ici, la valeur du paramètre est fournie à l'identificateur entier local, nombre, à partir du « point d'activation ». Ainsi :

```
WriteLn (Odd (carré (N DIV 100)))
```

qui imprimerait toujours faux, effectue trois opérations :

1. L'expression N DIV 100 est calculée.
2. Cette valeur entière temporaire est fournie à carré comme paramètre de valeur.
3. Son carré est renvoyé et fourni à Odd.
4. Le résultat booléen est maintenant évalué et fourni à la procédure WriteLn comme un autre paramètre de valeur.

Si N avait eu la valeur 17 lorsque l'instruction ci-dessus était exécutée, quelle aurait été ensuite sa valeur? Cette question peut sembler ridicule, évidemment. Il n'y a aucune raison pour que la valeur N change pendant l'évaluation d'une autre fonction impliquant N comme paramètre. Les résultats de toutes les fonctions ne doivent dépendre que de l'état des choses telles qu'elles existent. C'est-à-dire que la valeur renvoyée est fonction des valeurs de ses arguments ou paramètres. Le mécanisme PASCAL qui permet de fournir uniquement la valeur des variables garantit que, même si les paramètres sont changés localement (et parce qu'ils ne sont en fait que des copies locales), les véritables paramètres ne seront pas modifiés au point d'activation.

Pour ces raisons, bien que PASCAL ne l'interdise pas, on ne devrait pas avoir accès globalement aux données. Toutes les liaisons entre les appels de procédures et de fonction doivent être commandées par des listes de paramètres — même si les données se situent à l'intérieur du domaine du sous-programme. La seule exception à cette règle générale concerne l'accès à des constantes globales. En raison de leur nature, il est tout à fait impossible de modifier leurs valeurs. Cependant, si une valeur constante est fournie comme paramètre, elle devient une variable locale et n'est donc plus immuable.

```
FUNCTION Minuscule (character:char) :char;
  {renvoie la minuscule de tout argument majuscule}
  CONST
       décalé = 32; {ASCII ord ('a') – ord ('A')}
  BEGIN
  IF character IN ['A'. .'Z']
     THEN
        Minuscule:= char ( ord(character) + décalé)
     ELSE
        Minuscule:= character
  END;{Minuscule}
```

Lorsque nous définissons des procédures, il est quelquefois essentiel qu'elles modifient les valeurs de leurs paramètres — sinon elles n'exécuteraient pas la tâche pour laquelle elles ont été conçues. La procédure PASCAL read en donne un bon exemple. Il ne serait pas très pratique si read (N) ne donnait qu'une valeur à une variable entière locale à read et si la valeur de N demeurait inchangée. Dans de telles situations, nous devons fournir l'adresse d'un paramètre variable (et non sa valeur) afin que la procédure puisse y référer directement plutôt qu'à une copie locale. Ce mécanisme se nomme « fournir par adresse » ou « par référence » et ne doit être normalement utilisé que par les procédures et non par les fonctions.

Pour fournir des paramètres variables, il suffit simplement d'inclure le mot réservé VAR avant tout item compris dans la liste de paramètres de la procédure. La syntaxe d'une liste de paramètres est en quelque sorte semblable à la déclaration VAR d'un bloc; mais au lieu de n'apparaître qu'une fois comme délimiteur, il n'apparaît que devant un élément qui doit être modifié par une procédure. Par exemple :

```
PROCEDURE Process (VAR Compteur :ComptListe);
```

Illustrons cette technique puissante en mettant au point un programme qui lit quelques noms au clavier et, associé à chaque nom, un montant d'argent que chaque personne nous doit. Pour simplifier, nous n'utiliserons qu'une chaîne par nom et exprimerons les dettes sous forme de francs. Dès que notre programme est testé, nous pouvons ajouter d'autres détails comme les adresses, les numéros de téléphone et ainsi de suite. Nous aimerions imprimer la liste des noms soit par ordre alphabétique, soit, si possible, dans un ordre croissant de sommes dues. La structure naturelle de données est une liste d'enregistrements renfermant des zones de type approprié pouvant facilement être ajoutées ultérieurement. Nous pouvons manipuler chaque enregistrement comme un élément unique de donnée, mais utiliser toute zone comme clé de tri. Nous devons concevoir la représentation des données :

```
CONST
  LongueurChaîne  =20;
  LongueurListe   =50;

TYPE
  Cardinal        =0..MaxInt;
  TailleChaîne    =1..LongueurChaîne;
  Chaîne          =PACKED ARRAY [TailleChaîne] OF char;
data              =RECORD
                     nom : chaîne
{autre zone à adapter} debt : Cardinal;
                     END;{données}
  débits          =1..LongueurListe;
  EnregistrementListe=ARRAY [débits] OF données;
```

Cela permet l'utilisation de cinquante noms et suppose (pour maintenant) qu'aucun nom ne comptera plus de vingt caractères, mais en utilisant des définitions, nous pouvons changer ceci. Nous avons exprimé toutes les fourchettes et les structures dans une partie de définition TYPE pour plusieurs raisons :

1. La sécurité — un index de type bounds ne peut jamais excéder les limites définies d'un tableau, par exemple.
2. La mise au point — si une « erreur de fourchette » survenait lors des tests, le message d'erreur faciliterait la localisation du problème.
3. L'efficacité — les variables locales peuvent être de « types existants », ce qui évite la duplication et économise de la mémoire.
4. La nécessité — PASCAL exige que les indicateurs de type de paramètre soient des identificateurs et non des constantes (0...255, par exemple).

Ensuite, nous devons songer à concevoir quelques variables essentielles et un algorithme.

Avantages

Le programme CardReader manipule des nombres positifs d'une manière très sûre en lisant les données comme des caractères. La conversion en une valeur numérique équivalente est effectuée simplement en soustrayant la valeur d'origine du caractère chiffre de celle du chiffre 0. Ceci fonctionnera pour tous les jeux de caractères puisque les caractères 0 à 9 sont définis comme étant contigus. Notez que le programme n'a que deux variables globales (au niveau 0).

Les données de traitement des nombres et la fonction Value, qui se charge d'établir les correspondances, ne sont utilisées que par ReadCard, et ne sont donc déclarées et définies qu'à l'intérieur de cette procédure.

Il n'y a qu'un problème dans ce programme : si une valeur supérieure aux entiers prévus (MaxInt) est entrée, l'instruction d'affectation dans la boucle WHILE bloquera l'exécution.

```
PROGRAM        LecteurdeCarte  ( entrée, sortie );

TYPE
      cardinal          = 0 .. MaxInt;

VAR
      PosNum   : cardinal;
      Okay     : booléen;

      {11111111111111111111111111111111}

PROCEDURE    LireCarte  ( VAR N    : cardinal;
                          VAR OK   : booléen );
  CONST
    espace   = ' ';
  TYPE
      single  = 0 .. 9;
  VAR
      symbole  : char;
      nombres  : SET OF char;
            {22222222222222222222222}
                  {niveau 2}
  FUNCTION  Valeur ( nombre  : char )   : single;
        {Retourne la valeur numérique d'un
        chiffre sur tout jeu de caractère}
    BEGIN
      Valeur := ord ( nombre ) - ord ( '0' )
    END;      {valeur}
            {22222222222222222222222}
                  {de retour au niveau 1}
  BEGIN   {LireCarte}

    REPEAT
      lire ( symbole)

    UNTIL symbole > espace

    nombres  := [ '0' .. '9' ];

    OK := symbole IN nombres

    IF OK THEN
      BEGIN
        N := 0;

        WHILE symbole IN nombres  DO
          BEGIN {calcule la base 10}
            N := 10 * N + Valeur(  symbole);
            lit  ( symbole)
          END  {tout caractère délimiteur ou 'EoLn'}
      END      {après le nombre est supprimé}

  END;  {LireCarte}
        {11111111111111111111111111111111}
BEGIN   {LecteurCarte - Programme principal niveau 0}

  page (  sortie );
  WriteLn;
  WriteLn ( '  Ce programme lit des entiers positifs );
  WriteLn ( dans la fourchette 0..',
              MaxInt : 1, ' ) );
  WriteLn ( La fin du programme est' );
  WriteLn ( Contrôlée par détection d'erreur' );
  WriteLn;
  Write ( ' Entrez un nombre : ' );
  ReadCard ( PosNom, okay );

  WHILE okay DO
    BEGIN
      WriteLn;
      Write ( 'Le nombre entré est : ');
      WriteLn ( PosNom : 1 );
      Write ( 'Nombre? ' );
      ReadCard ( PosNom, okay
    END;

  WriteLn ( '---  ERREUR DETECTEE  ---' : 40 )
END .
```

Entrées multiples

Nous allons donner plusieurs exemples qui mettent en œuvre des bases de données « unidimensionnelles » classiques.

Un enregistrement est, dans une base de données conventionnelle, presque entièrement autonome : il contient toute l'information nécessaire, sans qu'il soit besoin de se référer à un autre élément. Les fiches cartonnées des bibliothèques sont de ce type. Chacune d'elles se réduit à la juxtaposition de certaines zones : TITRE, AUTEUR, ÉDITEUR et ISBN (International Standard Book Number; numéro d'identification). Il n'est nul besoin de chercher plus loin.

Parfois il arrive qu'une zone donnée contienne une entrée qui constitue un enregistrement à elle seule. C'est ainsi que dans le fichier des membres d'une association, ENFANTS peut renvoyer à d'autres enregistrements : si les fils d'un membre sont eux-mêmes membres, ils auront droit à une fiche spécifique. Mais une zone peut aussi faire référence à des enregistrements qui ne s'intègrent pas bien dans la structure générale du fichier. Considérons l'exemple d'une base de données intitulée PIÈCES, et qui est divisée en quatre zones : NUMÉRO, PRIX, STOCK et FOURNISSEUR. Il est plus que probable que FOURNISSEUR renverra à d'autres enregistrements qui ne pourront être répartis dans la base de données telle qu'elle a été définie.

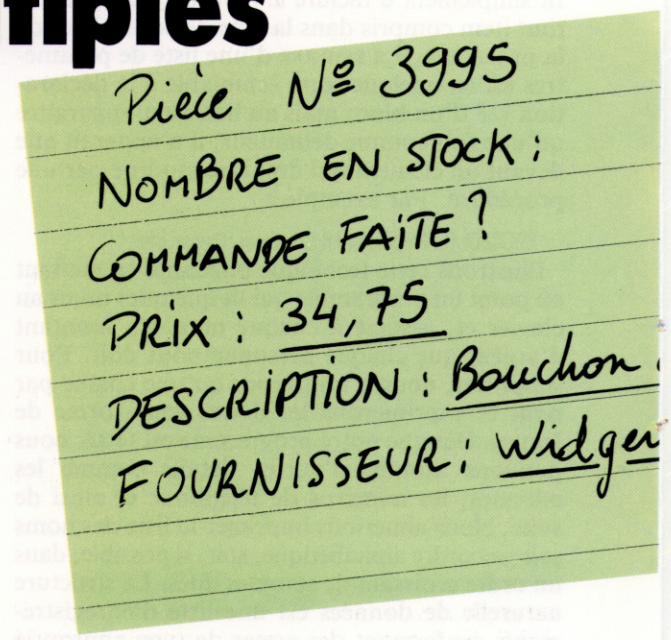
Pour illustrer ce point, voici tout d'abord le fichier MEMBRES DE L'ASSOCIATION :

NOM	Pichegru
ANNÉE D'INSCRIPTION	1979
COTISATION	Réglée
PROFESSION	Enseignant
SALAIRE	90 000
ENFANTS	Jacques; Sylvie

La zone ENFANTS contient dans cet exemple deux entrées, et les deux enfants pourraient bien sûr faire partie du même fichier à titre personnel, s'ils étaient aussi membres de l'association. Comparons cela à la base de données PIÈCES :

NUMÉRO	3995
PRIX	34.75
STOCK	86
DESCRIPTION	Bouchon de carburateur
FOURNISSEUR	Widgerama Ltd.; Dongle Corp. de Taiwan

Là encore, il y a deux entrées dans la zone FOURNISSEUR; mais aucune d'entre elles ne pourrait se voir accorder un enregistrement particulier. Il faudrait donc que la base de données comporte un second fichier réservé aux fournisseurs, dans lequel chaque fiche aurait une allure de ce type :

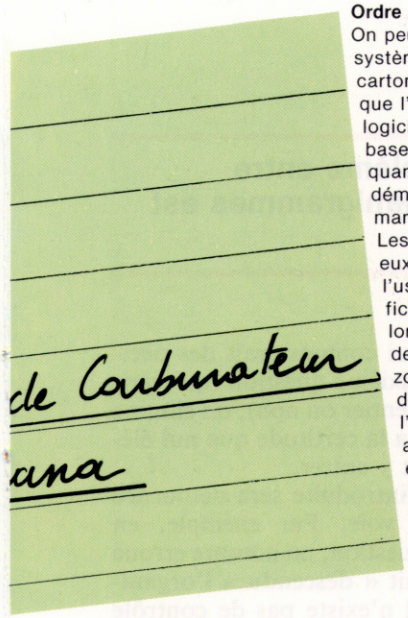


FOURNISSEUR	Dongle Corp., Taiwan
ADRESSE	57 Kau Moo Road, Taipei, Taiwan
TÉLÉPHONE	010-886-2-223-4478
FOURNITURES1	Bouchon de carburateur
PRIX 1 ($US)	12.45
FOURNITURES2	Antigel (bidon 1/2 litre)
PRIX 2 ($US)	6.32
FOURNITURES3	Poudre à polir (petit bidon)
PRIX 3 ($US)	3.27
FOURNITURES4	Poudre à polir (grand bidon)
PRIX 4 ($US)	6.11
FOURNITURES5	–

Il est bien évident que les informations dont nous avons besoin, s'agissant des fournisseurs, sont tout à fait différentes de celles relatives aux pièces qu'ils fournissent. La seule solution consiste donc à créer deux bases de données : un fichier pièces, un autre consacré à ceux qui les distribuent.

Si l'on veut toutefois s'en servir de façon productive, il est clair que le logiciel de gestion de l'ensemble devra être capable de gérer simultanément plus d'un fichier. Les programmes les plus simples n'en sont malheureusement pas capables, mais d'autres, comme dBase II, peuvent se servir d'un fichier (ainsi PIÈCES) comme fichier « primaire » et d'un autre (disons FOURNISSEURS) comme fichier « secondaire ». Si le logiciel est particulièrement sophistiqué, il doit pouvoir effectuer un tri sur le premier (pour en extraire des enregistrements) et aussi rechercher les enregistrements nécessaires (FOURNISSEURS) dans les fichiers apparentés.

dBase II permet l'emploi simultané de deux fichiers, qu'il associe en leur attribuant des clés communes. Deux commandes permettent le passage des références de l'un à l'autre : SELECT PRIMARY et SELECT SECONDARY. Si le fichier primaire est PIÈCES et le second FOURNISSEURS, alors que nous sommes en train de travailler sur le fichier PIÈCES,



Ordre et progrès
On peut mettre au point un système de fiches cartonnées aussi complexe que l'on voudra. Les logiciels de gestion de bases de données doivent, quant à eux, adopter une démarche structurée pour manipuler les informations. Les plus élaborés d'entre eux doivent permettre à l'usager d'appeler des fichiers reliés entre eux, et, lorsqu'il entre des données, de supprimer certaines zones rendues inutiles par d'autres éléments de l'enregistrement. Une pièce automobile ne sera par exemple commandée que si le stock est en dessous d'un certain niveau (ou s'il est déjà tombé à zéro). (Cl. Kevin Jones.)

nous pourrons faire usage de la commande USE PIÈCES. Si à partir de là nous voulions faire des références croisées au fichier des fournisseurs, nous taperions SELECT SECONDARY suivi, ligne suivante, de USE FOURNISSEURS. Les nombreuses commandes du programme seraient à notre disposition et s'appliqueraient au fichier sélectionné en dernier lieu.

Il arrive parfois que le contenu d'une zone particulière (ainsi FOURNISSEURS dans le fichier PIÈCES déjà étudié) n'ait pas besoin d'être individualisé (organisée en un fichier séparé). Les différentes zones d'un enregistrement devraient suffire dans ce cas. Supposons que vous ayez créé une base de données consacrée à la technologie automobile et à ses récents développements. Vous y classeriez des articles de presse et des références de livres consacrés, entre autres, aux carosseries en plastique moulé ou aux alimentations à injection. Le format de base pourrait être celui-ci :

```
SUJET
RÉSUMÉ1
RÉSUMÉ2
RÉSUMÉ3
SOURCE      ISBN
TITRE
DATE
PAGE
```

Si vous avez trouvé la référence dans une revue, le numéro d'ISBN ne sera d'aucune utilité puisque les magazines en sont dépourvus. Si, par contre, il s'agissait d'un livre intitulé *la Programmation des microprocesseurs pour la conduite automobile intégrée,* vous auriez besoin de cette indication. De même, vous auriez un numéro de page dans les deux cas. Cela n'aurait pas lieu d'être, si vous aviez trouvé le renseignement à l'occasion d'une émission de télévision sur TF1. Certains programmes, comme Rescue, de Microcomputer Systems, permettent l'absence ou la présence de telle ou telle zone dans un enregistrement particulier qui dépend d'autres champs précédemment définis.

C'est ainsi que si SOURCE n'est pas un livre, le numéro d'ISBN ne sera pas pris en ligne de compte, et donc ne sera pas affiché sur l'écran comme sur l'imprimante. Si cette source n'est ni un livre ni une revue, PAGE sera mis à l'écart. Un logiciel de gestion capable de manœuvres de ce genre peut ainsi économiser de l'espace mémoire, en éliminant les zones inutiles. Cela permet aussi d'améliorer la présentation. Toutefois, il restera moins souple d'emploi qu'un programme capable de relier des enregistrements répartis sur deux fichiers différents.

Il se peut qu'une base de données comporte des informations qui restent constantes pour chaque enregistrement, ainsi que certaines données qui peuvent, ou non, être mises en œuvre; c'est ce qu'on appelle une « hiérarchie à deux niveaux ». En règle générale, les logiciels de gestion multifichiers traitent les renseignements de ce type comme un cas particulier, ou un sous-ensemble, d'une base de données à fichiers multiples. Une référence à un livre (SOURCE) pourrait renvoyer à un fichier LIVRES, par exemple.

Il reste à considérer un dernier élément. Vous pouvez mettre au point un système de fiches cartonnées qui sera aussi complexe et sophistiqué que vous le voudrez. Mais la plupart des gestionnaires de base de données ont des limites, selon ce qu'ils peuvent ou ne peuvent pas accepter. C'est pourquoi, avant d'en acquérir un pour votre micro-ordinateur, il est très important d'avoir une idée précise de ce que vous comptez lui demander. Rassemblez vos exigences et examinez de près les caractéristiques de chaque programme avant de prendre une décision définitive.

Quelques bases de données

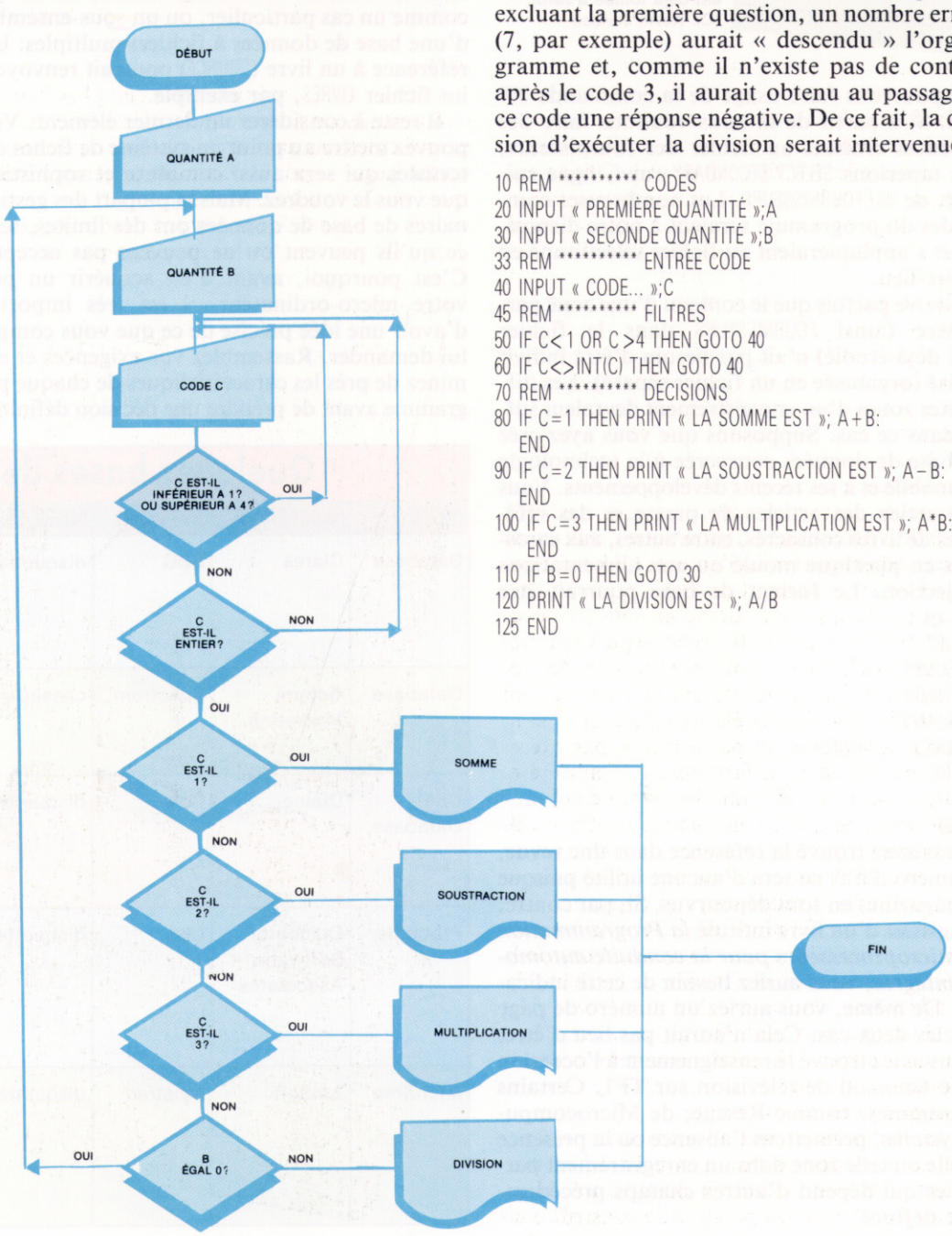
NOM	FABRICANT	APPAREIL	FORMAT	COMMENTAIRES
Betabase	Clares	BBC	disquette	Long max. enregistr. 2048 Taille max. fichier 99 K (40 T), 199 K (80 T) Max. champs 200.
Database	Gemini Marketing	Spectrum	cassette	Système à index. Enregist. définis par usager.
DFM Database	Dialog	C64	disquette	15 zones/enregistr. Zone alphanum. 36 car. Zone numérique 9 chiffres.
Practifile	Computer Software Associates	C64	disquette	3 800 enregistr./fichiers Réorganisation données pour courrier électronique. Entrée par lots.
MicroPen	AMSoft	Amstrad	disquette	Contient traitement de texte et tableur. Long. max. enregistr. 1024. Nb max. d'enregistr. par fichier 32750.

Fin de cascade

Nous terminons ce type de test par un nouveau problème entre variables et constantes. Comme quoi l'étude des organigrammes est importante... et pas seulement pour les débutants.

Dans ce deuxième organigramme, la dernière question de la série éliminatoire d'opérations à réaliser a été supprimée. Ainsi, l'on peut constater l'introduction d'une comparaison, pour savoir si un nombre donné se situe entre un et quatre ; néanmoins, il existe un autre problème : qu'arriverait-il si l'on introduisait, à ce même niveau, un nombre qui comporterait des décimales ? Grâce à la deuxième question (qui vérifie si le nombre est un entier ou non), on élimine cette possibilité et on a la certitude que nul élément étranger ne peut y entrer.

Toute information introduite sera détournée sur une quelconque voie. Par exemple, en excluant la première question, un nombre erroné (7, par exemple) aurait « descendu » l'organigramme et, comme il n'existe pas de contrôle après le code 3, il aurait obtenu au passage de ce code une réponse négative. De ce fait, la décision d'exécuter la division serait intervenue.




```
10 REM ********** CODES
20 INPUT « PREMIÈRE QUANTITÉ »;A
30 INPUT « SECONDE QUANTITÉ »;B
33 REM ********** ENTRÉE CODE
40 INPUT « CODE... »;C
45 REM ********** FILTRES
50 IF C<1 OR C>4 THEN GOTO 40
60 IF C<>INT(C) THEN GOTO 40
70 REM ********** DÉCISIONS
80 IF C=1 THEN PRINT « LA SOMME EST »; A+B:
   END
90 IF C=2 THEN PRINT « LA SOUSTRACTION EST »; A-B:
   END
100 IF C=3 THEN PRINT « LA MULTIPLICATION EST »; A*B:
    END
110 IF B=0 THEN GOTO 30
120 PRINT « LA DIVISION EST »; A/B
125 END
```

Kit pour six robots

Fischertechnik vous propose de construire avec ses éléments en kit six types de robot, ou de concevoir votre propre modèle. Cela vous garantit plusieurs heures de travail et de découvertes.

Bien que de nombreux spécialistes de l'informatique grand public prédisent depuis un certain temps que l'avenir de l'industrie se situe dans le développement de la robotique domestique, ces prévisions ne se sont pas toujours révélées exactes. Ce n'est pas difficile à comprendre. Malgré le fait qu'au cours de la dernière année de nombreux « robots domestiques » soient apparus sur le marché, ils ont présenté quelques problèmes importants. Ces difficultés ont compromis l'enthousiasme que ce projet aurait dû susciter auprès du public.

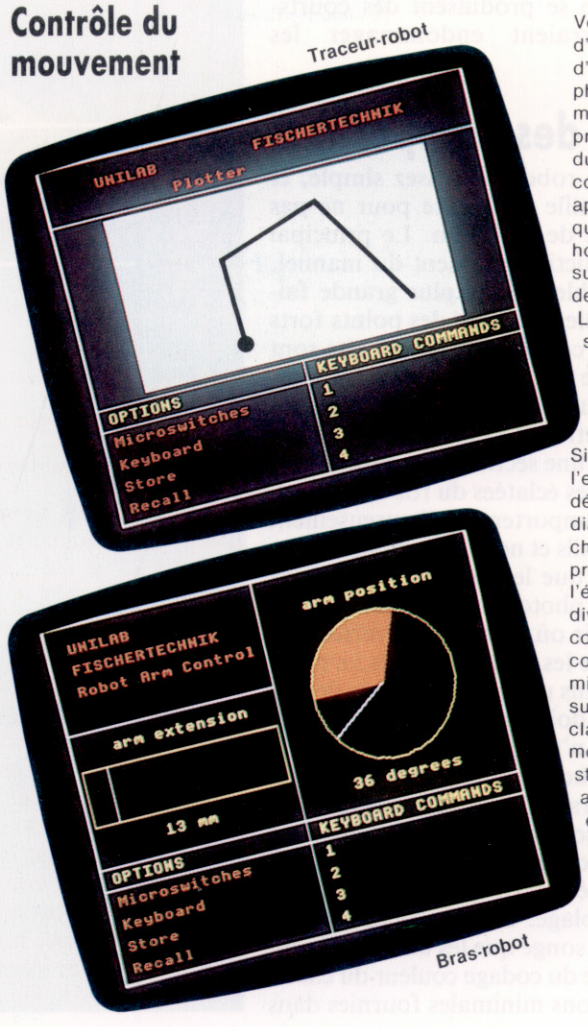
Il existe d'une part des robots faciles à construire, comme les Movits, qui peuvent être assemblés en quelques heures par une personne qui ne connaît rien en robotique ou même en électronique. Ces unités ont cependant des applications très limitées, et elles ne suscitent pas un intérêt très durable chez l'utilisateur. D'autre part, des robots plus coûteux sont disponibles mais nécessitent généralement une connaissance étendue de la robotique et de l'électronique pour les utiliser. Même dans ce cas, plusieurs de ces robots évolués n'ont qu'un seul usage. De nouveau, cela signifie que l'utilisateur abandonne rapidement ce qui est devenu un gadget inutile.

Le robot idéal serait donc facile à construire et doté de différentes fonctions que l'utilisateur pourrait expérimenter facilement. Le kit robotique de Fischertechnik est censé réunir ces caractéristiques. L'approche des concepteurs de ce produit est très simple. Il existe depuis maintenant quelques années de nombreux kits électroniques destinés aux enfants qui leur permettent de construire des circuits électroniques assez simples pour assembler un poste radio, par exemple. Le même kit peut être ensuite démonté et reconstruit pour former un dispositif de chronométrage électronique ou une alarme contre le vol. Simultanément, des générations d'enfants ont grandi en s'amusant avec des kits Lego qui peuvent être utilisés et réutilisés pour construire une quantité énorme de modèles différents à partir des mêmes petits blocs en plastique. Lors de la conception de son kit électronique, Fischertechnik a adopté ces deux approches et les a associées à la technologie informatique moderne pour mettre au point un produit qui pourrait réussir la percée commerciale que de nombreuses personnes ont prédite pour la robotique.

Le kit Robotics permet à l'utilisateur de construire de nombreuses unités robots différentes pouvant être commandées à partir d'un micro domestique. Il est ainsi possible de construire un bras robot, une machine qui trie des articles de diverses longueurs, un traceur et un dispositif d'entrée graphique. Dès que l'utilisateur connaît à fond un robot, il peut le démonter et en reconstruire un autre. Afin de commander le robot à partir d'un ordinateur, une interface est nécessaire pour transformer les signaux numériques provenant de l'ordinateur en des signaux qui peuvent servir à actionner les moteurs électriques du kit.

Formés de nombreuses pièces en plastique, les composants du kit de construction peuvent être emboîtés les uns dans les autres — un peu comme un kit Lego — pour constituer différents modèles. Ces pièces prennent l'aspect de blocs qui peuvent être assemblés pour former les bras, ou utilisés pour abriter les unités de commande. Il y

Contrôle du mouvement



Voici les écrans d'introduction du logiciel d'interface Unilab. La photographie inférieure montre un affichage du programme de commande du bras-robot. Dans le coin supérieur droit apparaît un diagramme qui illustre le mouvement horizontal du robot. On suppose que la position de départ du robot est 0. Lorsque le bras-robot se déplace vers la droite (la ligne se déplace dans le sens des aiguilles d'une montre), le nombre de degrés augmente. Similairement, lorsque l'extension du bras est déplacée, la ligne du diagramme et la distance changeront dans la même proportion. Au bas de l'écran apparaissent les diverses options de commande. Celles-là sont commandées au moyen de micro-interrupteurs situés sur la carte ou à partir du clavier. Des séquences de mouvement peuvent être stockées dans l'ordinateur, appelées ultérieurement et réexécutées. Le même système de commande est utilisé avec le traceur-robot.

a aussi d'autres pièces, comme des roues d'engrenage et des vis d'entraînement. Une grande carte en plastique mesurant 260 mm sur 187 sert de base pour le robot lui-même ou, si un traceur ou une tablette graphique sont intégrés, comme support pour le papier. Les commandes électriques et le système d'alimentation doivent aussi être assemblés séparément à partir de divers composants. Le kit comprend de nombreuses prises mâles et femelles, huit dispositifs de commutation et une paire de moteurs électriques. On trouve également environ 1 mètre de câble-ruban à 20 voies et un bout de câble supplémentaire pour construire les connexions internes.

Dans l'ensemble, les pièces sont assez robustes et semblent promettre une longue durée d'utilisation, à l'exception peut-être des potentiomètres. La plaque de base de l'un de ceux-ci a été abîmée lors de la construction, et bien que cela ne gêne pas le fonctionnement du bras (voir la photographie), puisque la plaque de base a tout de même pu être installée, cela met en doute la fiabilité à long terme. Une autre remarque, concernant les potentiomètres, est que, contrairement au reste des composants électriques qui peuvent être vissés dans des prises, les fils nus ne peuvent être fixés sur ces unités que s'ils sont enroulés autour des connecteurs. Nous recommandons donc de souder les joints de façon permanente si l'on prévoit de faire un usage intensif du robot afin d'éviter que ne se produisent des courts-circuits qui pourraient endommager les potentiomètres.

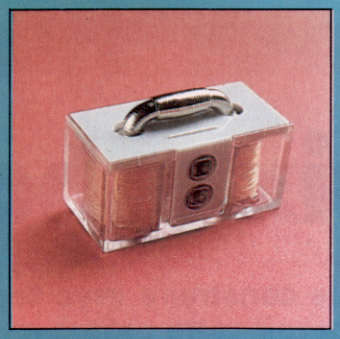
Installation des composants

La construction des robots est assez simple, et les pièces sont de taille suffisante pour ne pas avoir besoin d'outil de précision. Le principal problème de construction provient du manuel, qui constitue probablement la plus grande faiblesse du système. Bien qu'il ait des points forts — par exemple, les connexions de circuits sont faciles à comprendre —, le manuel renferme des photographies de qualité médiocre des modèles en cours de construction. Les photographies des pièces requises pour une section particulière sont accompagnées de vues éclatées du robot en construction. Elles ne comportent malheureusement pas beaucoup de détails et ne sont pas assez annotées, ce qui signifie que les utilisateurs devront parfois examiner les photographies assez longuement avant de savoir où placer les diverses pièces. De plus, comme les photographies ne montrent le robot que dans un seul plan, vous devrez souvent deviner où doit être installée une pièce lorsque sa position n'est pas visible.

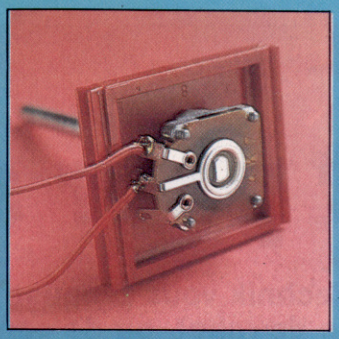
Pire encore, les photos sont en noir et blanc. Cela signifie que lorsque vous effectuez la dernière partie de la construction, qui consiste à monter le circuit électrique du robot, il est presque impossible de découvrir quelle est la véritable procédure de câblage. C'est particulièrement étrange lorsque l'on songe que les circuits dépendent en grande partie du codage couleur du câble-ruban. Les annotations minimales fournies dans les diagrammes de montage n'apportent guère d'aide, indiquant à peine, par exemple, que les fils E3 à E8 appartiennent à une section particulière.

Cela étant dit, nous devons souligner que le robot présenté ici a été construit par une personne n'ayant aucune connaissance en robotique ou en électronique, et qu'elle a réussi à faire fonctionner le robot. Il eût été préférable, toutefois, que Fischertechnik ne s'en soit pas remis au bon jugement des gens mais ait simplement fourni un meilleur manuel de construction.

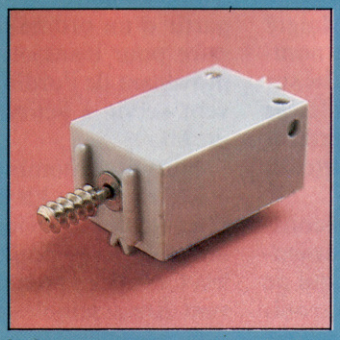
Lorsque la construction du robot est terminée, la tâche suivante consiste à le connecter à l'interface. Ici, heureusement, la procédure est beaucoup plus simple. La carte d'interface est composée de nombreuses prises divisées en sections servant à recevoir les conducteurs du câble-ruban.



Électro-aimant
La barre de métal située sur l'électro-aimant glisse à l'intérieur du dispositif de saisie du bras-robot. L'électro-aimant peut être mis en fonction et hors fonction, ce qui permet au robot de soulever les petites plaques métalliques fournies.

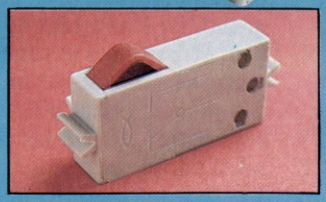


Potentiomètre
Les potentiomètres servent et fournissent une réaction et communiquent la position du robot.

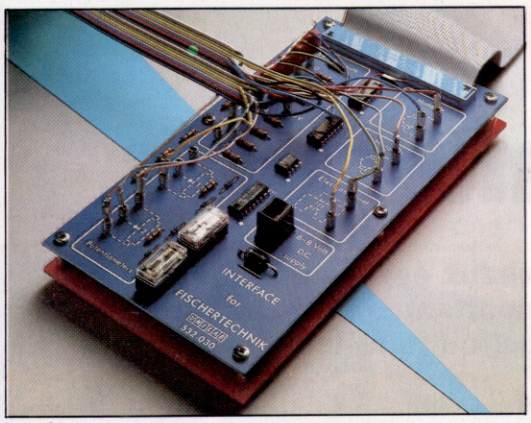


Le moteur
Le moteur est logé dans un boîtier en plastique. La connexion au moteur est établie au moyen de deux prises qui se trouvent sur le boîtier.

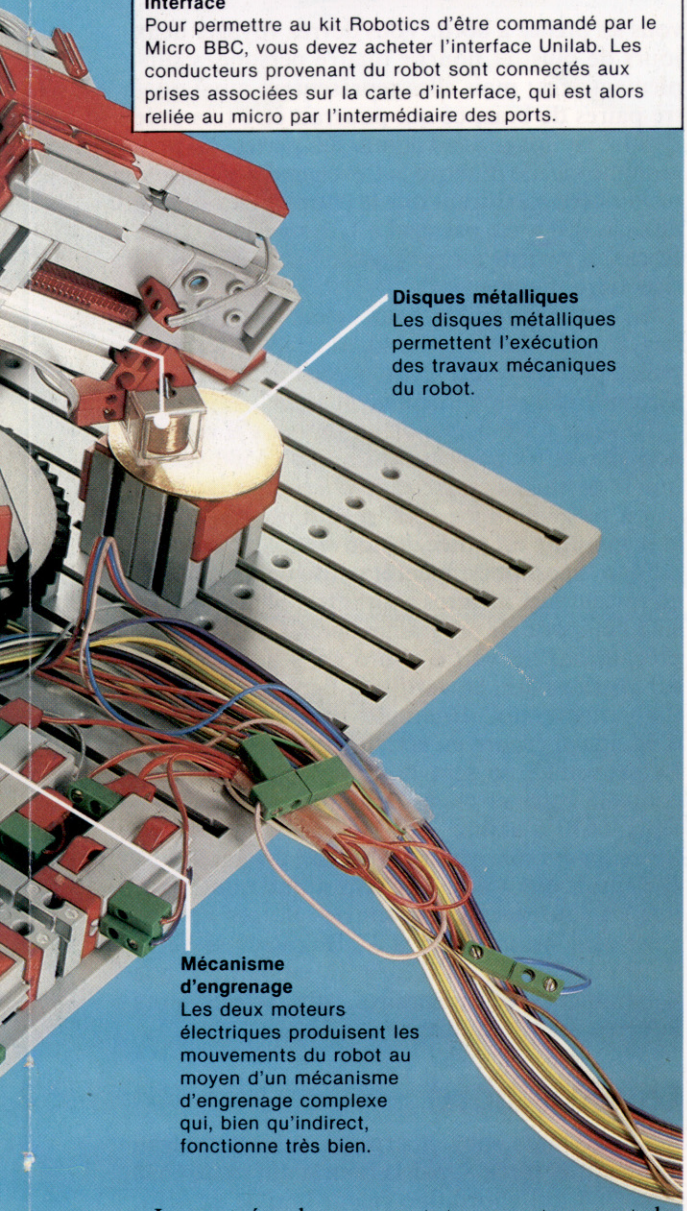
Voyants
Lorsque l'un des trois systèmes de commande principaux (électro-aimant, horizontal ou vertical) est activé, l'un des trois voyants s'allume pour indiquer à l'utilisateur quelle commande est utilisée.



Commutateur
En appuyant en même temps sur certains de ces commutateurs, un nombre est généré et est stocké dans le registre du port utilisateur, ce qui peut servir à commander directement le robot.



Interface
Pour permettre au kit Robotics d'être commandé par le Micro BBC, vous devez acheter l'interface Unilab. Les conducteurs provenant du robot sont connectés aux prises associées sur la carte d'interface, qui est alors reliée au micro par l'intermédiaire des ports.



Disques métalliques
Les disques métalliques permettent l'exécution des travaux mécaniques du robot.

Mécanisme d'engrenage
Les deux moteurs électriques produisent les mouvements du robot au moyen d'un mécanisme d'engrenage complexe qui, bien qu'indirect, fonctionne très bien.

Les entrées des commutateurs se trouvent du côté droit de la carte. La logique des entrées des commutateurs est commandée par le registre de données et par le registre de direction des données du micro ; son fonctionnement est similaire à celui du boîtier tampon que nous avons déjà construit. Plus bas, on retrouve une paire de potentiomètres qui permettent à l'ordinateur de contrôler l'alimentation fournie aux commutateurs et aux ampoules. Le contrôle des potentiomètres est assuré par l'intermédaire du port analogique.

De l'autre côté de la carte d'interface, on aperçoit des connexions servant à commander les moteurs et l'électro-aimant (ce dernier est utilisé par le bras-robot pour saisir des objets métalliques). Finalement, il y a une prise d'alimentation de 6 à 8 V, qui alimente le système au complet.

L'interface est connectée au micro *via* trois ports séparés — le port utilisateur, le port analogique et le port imprimante — utilisés pour commander les moteurs et l'électro-aimant. Puisque le système est destiné à des utilisateurs inexpérimentés, la carte d'interface a été construite de façon à éliminer la possibilité d'un câblage inadéquat qui pourrait conduire à appliquer aux interfaces de l'ordinateur des tensions pouvant endommager ses circuits.

Le manuel qui est fourni pour l'interface (écrit par le fabricant du produit, Unilab) est de bien meilleure qualité que le manuel de Fischertechnik Robotics. On y trouve une explication détaillée du montage de l'interface et de l'exécution du logiciel qui accompagne le système. On trouve également un bref résumé des adresses de commande utilisées, ce qui permet de commander les robots à partir de BASIC. Cependant, les diagrammes de montage semblent ne pas correspondre aux circuits Fischertechnik. Bien qu'une note relative aux divers câbles-rubans soit fournie avec l'interface, elle ne réussit pas, semble-t-il, à résoudre tous les problèmes et, de nouveau, vous devrez utiliser votre bon sens pour déterminer comment monter correctement le circuit.

Le langage prof

Une cassette renfermant trois programmes servant à faciliter la commande des robots accompagne l'interface. Les deux premiers sont des programmes spécialisés qui commandent le bras et le traceur, ce qui permet aux unités externes d'être commandées soit du clavier, soit des commutateurs de commande situés sur le robot lui-même.

Le troisième programme est Prof, un système d'exploitation général qui permet de commander directement les robots. Il s'agit essentiellement d'une extension du BASIC, comprenant quelques commandes supplémentaires. Plusieurs des mots introduits par Prof sont des noms de variables, comme Motor et Magnet, désignant les adresses de commande, tandis que d'autres sont introduits pour faciliter l'édition des séquences d'interventions devant être effectuées par les robots.

Bien que Prof soit assez complet et permette d'insérer des boucles conditionnelles et de concevoir une programmation structurée, il est limité principalement à la commande de la sortie. Par conséquent, le panneau graphique, une unité d'entrée, n'est pas pris en charge par Prof, ce langage ne prévoyant aucun stockage de l'information provenant de dispositifs d'entrée. Cela vous oblige à créer vos propres programmes en BASIC.

Malgré les problèmes posés par la documentation, le kit Robotics et son interface composent un système assez intéressant.

KIT FISCHERTECHNIK WAFARDRIVE

MOUVEMENT

Le robot est actionné par un ou deux moteurs CC.

COMMANDE

Soit à partir du clavier de l'ordinateur ou par l'intermédiaire des huit commutateurs de la carte du robot elle-même.

SOURCES D'ALIMENTATION

Une seule pile 6 V PJ996/4R25 (non comprise).

Problèmes multiples

Nous allons mettre en œuvre six nouveaux événements de premier plan, qui auront des conséquences importantes : rencontrer et sauver les occupants d'un canot de sauvetage, ou faire face à l'épidémie.

Intéressons-nous donc à la programmation de ces six événements majeurs, dont chacun peut se produire une fois au cours du voyage. Nous disons « majeurs », parce qu'ils sont bien plus lourds de conséquences que ceux que nous avons déjà introduits dans notre programme. L'un d'eux sera sélectionné au hasard, au début de chaque semaine, sauf si, bien sûr, il s'est déjà produit auparavant : dans ce cas, il ne se passera rigoureusement rien.

La ligne 49 permet de DIMensionner le tableau M(), qui servira de drapeau en signalant si l'un des six scénarios a été choisi une première fois. On passe au sous-programme de la ligne 6500, qui procède à un tirage au sort à partir d'un GOSUB du programme principal (ligne 870). La ligne 6508 détermine un nombre aléatoire compris entre 1 et 10. Comme il y a six événements possibles, il y a donc 60 % de chances que l'un d'eux soit sélectionné. Si, par contre, 7, 8, 9 ou 10 sont choisis, il ne se passera rien. On peut naturellement modifier les pourcentages de départ, de façon à accroître ou à diminuer les probabilités. Si, par exemple, on réécrit la ligne 6508 de la façon suivante :

```
X=INT(RND(1)*8)+1
```

les chances d'avoir affaire à un événement majeur passent de 6 à 8, c'est-à-dire à 75 %. Toutefois, il faut tenir compte du fait qu'une fois que l'un d'eux a été choisi, ce qui a entraîné la création du drapeau correspondant au sein du tableau M(), la probabilité de sélection d'un des cinq autres événements tombe à 5 sur 8, soit 62,5 %. Pour un troisième, on passe à 4 sur 8, soit 50 %, et ainsi de suite. Plus il se produit de choses, moins il risque de s'en produire davantage...

C'est l'instruction ON X GOSUB qui assure l'« aiguillage » du programme. Elle est utilisée à peu près de la même façon que ON X GOTO, qui choisissait les événements mineurs. Si X, le nombre aléatoire, est égal à 1, on passe au premier des numéros de lignes spécifiés après l'instruction elle-même. Ce sera ici le sous-programme 6530 qui vous fait croiser un canot de sauvetage. Si X est égal à 2, le sous-programme de la ligne 6700 (la maladie mortelle) est parcouru. Nous nous en tiendrons là pour cette fois, mais à l'avenir nous devrons ajouter, ligne 6510, les quatre adresses restantes.

Dans le premier sous-programme, vous apercevez à travers votre télescope un canot de sauvetage à bord duquel se trouvent quatre hommes et un grand coffre. C'est à vous de voir si vous les prenez à bord. Vous devrez voyager deux jours de plus, et nourrir quatre personnes supplémentaires ; mais elles représentent aussi quatre paires de bras, ce qui pourrait être très utile si votre équipage avait connu des pertes. Le programme vérifie d'abord si l'événement s'est déjà produit (en s'assurant que le premier élément du tableau MC() n'est pas égal à 1, ligne 6535). Dans ce cas, il revient au programme principal, sans rien faire d'autre.

Si, par contre, il a pour valeur zéro, M(1) est mis à 1 par la ligne 6540, de façon qu'il ne puisse plus survenir. Cette procédure est respectée pour chaque événement majeur.

La ligne 6576 demande au joueur s'il veut ou non sauver les naufragés. La ligne 6580 vérifie que le premier caractère de la réponse est O ou N ; on retourne ligne 6568 si ce n'est pas le cas, et la question est posée de nouveau. Si la réponse est non, le canot est bientôt perdu de vue, et l'on en revient aux affaires courantes. Si la compassion l'emporte, le programme prend toutefois des précautions : il voit d'abord s'il y a de la place sur votre navire, en vérifiant, ligne 6625, que CN (la variable qui décompte tous les membres d'équipage encore en activité) est bien égal à 16. Le sauvetage ne sera donc pas possible, et le contrôle repasse au programme principal. La ligne 6630 s'assure ensuite que vous pouvez accueillir les survivants : X est fixé à 16 moins CN. Il s'ensuit que X doit être plus grand que 3 pour que les naufragés soient sauvés. Si, par contre, X est égal ou inférieur à 3, le joueur se verra informer du fait qu'il ne peut embarquer que X hommes supplémentaires, qui seront alors intégrés dans l'équipage.

La vie à bord

Les survivants sont incorporés dans le tableau « catégorie/force », par la boucle qui va de 6638 à 6697. X a d'abord une valeur initiale de zéro, et sert à décompter les matelots supplémentaires. Le tableau TS(,) est parcouru pour voir si la catégorie est égale à zéro, ce qui signifie qu'il y a de la place pour un homme de plus. En cas de recherche positive, X est incrémenté de 1 (ce qui marque l'ajout d'un homme d'équipage). Quand X atteint 4, tous les survivants ont été enrôlés au sein de votre main-d'œuvre. La ligne 6655 fixe donc à 16 la valeur de T, si bien que le programme quitte la boucle dès qu'il atteint NEXT. S'il n'est pas possible de prendre quatre personnes de plus, il fait de même après que le

Ian McKinnell

tableau a été vérifié à seize reprises. Ceux des rescapés qui ont pu monter à bord sont ajoutés à CN, qui tient le décompte des hommes. Chacun d'eux reçoit, à l'intérieur du tableau TS(,), une valeur aléatoire : entre 1 et 5 pour la catégorie, entre 50 et 99 pour la force. Les nouveaux venus sont donc en bonne ou en très bonne santé.

Le coffre contenait par ailleurs des provisions. Une boucle est créée ligne 6685 pour chacun des quatre types de provisions, tandis que la ligne 6690 génère un nombre aléatoire compris entre 10 et 19, ce qui représente la quantité de chacune d'elles. La ligne 6692 affiche ce chiffre, ainsi que les unités, gérées par U$(), qui représente des kilos ou des barriques. Si, pendant la semaine, telle ou telle provision a été épuisée, la ligne 6693 modifie la quantité correspondante de -999 à 0, pour qu'il soit tenu compte de l'apport, et la ligne 6694 l'ajoute à ce qui était déjà à bord. A noter aussi que, la répartition des survivants par catégories étant faite de façon aléatoire, un navire privé de médecin peut bénéficier de la présence d'un docteur à bord du canot de sauvetage.

La peste

Le second événement majeur n'est autre qu'une épidémie de peste, appelée ligne 6700 par le second numéro de ligne de l'instruction ON X GOSUB. L'impact de la maladie sera déterminé par deux facteurs : y a-t-il ou non un médecin à bord, et avez-vous pris la précaution d'acheter des flacons de médicament en début de partie? Tout comme pour le sous-programme précédent, le programme vérifie évidemment si tout cela ne s'est pas déjà produit pendant le jeu, en voyant si M(2), second élément du tableau M(), n'a pas déjà une valeur de 1 : si oui, il repasse en boucle principale sans que rien d'autre ne se passe.

Si l'événement est encore inédit, il faut d'abord constater la présence, ou l'absence, d'un médecin à bord du navire. C'est pourquoi une boucle de 1 à 16 est créée pour voir si le tableau catégorie/force contient une valeur de 2 (indiquant qu'il y en a un). S'il n'est pas déjà mort, sa force est différente de 0 ou de -999, et X reçoit une valeur de 0. Si X, une fois la boucle achevée, est égal à 1, cela signifie qu'il n'y a pas de médecin. Après quoi, le programme cherche à savoir s'il y a des médicaments à bord. Pour cela, il vérifie la valeur de PA(1), l'élément correspondant dans le tableau des provisions. Si Y vaut zéro, il en existe. Dans le cas contraire, il sera égal à 1.

La ligne 6730 crée un facteur de maladie, Z. Il permet de calculer l'affaiblissement de l'équipage. Z est en effet égal à ((X+Y)*10)+5)). X et Y sont bien sûr fonction de la présence, ou de l'absence, de médecin et de remèdes. La force sera donc réduite de 25 si ces deux éléments sont manquants, de 15 si un seul manque, de 5 s'ils sont tous deux présents. Les lignes 6734 et 6736 précisent au joueur les raisons de la sévérité de l'épidémie, et vérifient la présence du médecin et des remèdes en se servant des valeurs de X et de Y définies précédemment.

Si certains membres d'équipage étaient déjà très malades, il est plus que vraisemblable que la maladie les tuerait. X est mis à zéro et utilisé pour décompter les morts. Une boucle allant des lignes 6755 à 6775 parcourt le tableau « catégorie/force » et réduit les forces de ceux qui sont touchés; en effet, tous les matelots ne sont pas atteints. La ligne 6756 génère un nombre aléatoire. S'il est inférieur à 0.3, la force du membre

d'équipage correspondant dans le tableau ne sera pas affectée par la maladie. Chaque homme a donc environ 30 % de chances d'échapper à l'épidémie : vous pourrez, si vous le désirez, modifier ce facteur ou lui donner à lui aussi une valeur aléatoire. La ligne suivante s'assure que le marin en question n'est pas déjà mort. S'il vit encore, sa force est réduite de Z unités. Il se peut que cela le tue, en la faisant descendre au-dessous de zéro. Une procédure de vérification la remet à -999. Lors du relevé hebdomadaire, elle sera extraite par le programme qui procédera aux modifications nécessaires. X, le nombre d'hommes tués par la peste, sera ainsi incrémenté de 1. Si d'aventure il y avait des remèdes (ce qui est indiqué par Y=0), la ligne 6776 en divise la quantité par deux et arrondit le résultat de façon à obtenir un nombre entier de bouteilles. Il se peut aussi que personne ne soit mort (X=0); dans ce cas, on revient à la boucle principale. Dans le cas contraire, le joueur apprendra combien de ses hommes ont péri.

Dans le module suivant, nous verrons quelques-uns des autres événements majeurs qui peuvent survenir au cours de la traversée, notamment de sérieux problèmes avec le gouvernail.

Module huit :
Deux événements majeurs

Dimension tableau-drapeau

```
49 DIMM(6):REM INDICATEURS POUR SAVOIR SI EVENEMENTS MAJEURS
DEJA SURVENUS
```

Addition à boucle principale

```
870 GOSUB6500:REM EVENEMENTS MAJEURS
```

S/P sélection événements majeurs

```
6500 REM EVENEMENTS MAJEURS
6508 X=INT(RND(1)*10)+1
6510 ON X GOSUB6530,6700
6520 RETURN
```

S/P événements majeurs

```
6530 REM CANOT SAUVETAGE
6535 IFM(1)=1THENRETURN
6536 PRINTCHR$(147)
6540 M(1)=1
6550 S$="UN CANOT DE SAUVETAGE+":GOSUB9100
6552 S$="APPARAIT AU LOIN+":GOSUB9100
6554 PRINT:GOSUB9200
6556 S$="AU TELESCOPE VOUS VOYEZ+":GOSUB9100
6558 S$="QU'IL CONTIENT+":GOSUB9100
6560 PRINT:GOSUB9200
6562 S$="4 PERSONNES+":GOSUB9100
6563 GOSUB9200
6564 S$="ET UN GRAND COFFRE!+":GOSUB9100
6566 PRINT:GOSUB9200
6568 S$="EN CHANGEANT DE CAP+":GOSUB9100
6570 S$="POUR LES SAUVER+":GOSUB9100
6572 S$="VOUS PERDREZ DEUX JOURS.+":GOSUB91
00
6574 PRINT:GOSUB9200
6576 S$="VOULEZ-VOUS LES SAUVER (O/N)?+":GOSUB
9100
6578 INPUTI$:I$=LEFT$(I$,1)
6580 IFI$<>"o"ANDI$<>"N"THEN6578
6585 IFI$="o"THEN6600
6588 PRINT:GOSUB9200
6590 S$="LE CANOT DISPARAIT AU LOIN...+":GOSUB9100
6592 PRINT:GOSUB9200
6594 S$=K$:GOSUB9100
6596 GETI$:IFI$=""THEN6596
6599 RETURN
6600 PRINT:GOSUB9200
6610 EW=EW+2/7
6625 IFCN<>16 THEN6630
6627 S$="VOUS NE POUVEZ LES SAUVER+":GOSUB9100
6628 S$="PAR MANQUE DE PLACE SUR LE BATEAU+":GOSUB9100
6629 GOTO6592
6630 X=16-CN:IFX>3 THEN6635
6632 S$="VOUS N'AVEZ DE PLACE QUE POUR+":GOSUB9100
6633 PRINT: "PERSONNES DE PLUS+"
6634 PRINT:GOSUB9200
6635 S$="VOUS SAUVEZ:+":GOSUB9100
6638 X=0
6640 FORT=1TO16
6645 IFTS(T,1)<>0THEN6679
6650 X=X+1
6655 IFX>4THENT=16:GOTO6679
6660 CN=CN+1
6665 TS(T,1)=INT(RND(1)*5)+1
6668 TS(T,2)=INT(RND(1)*50)+50
6670 PRINT"1 ";C$(TS(T,1))
6679 NEXT
6680 PRINT:GOSUB9200
6682 S$="LE COFFRE CONTIENT:+":GOSUB9100
6685 FORT=1TO4
6690 X=INT(RND(1)*10)+10
6692 PRINTX;U$(T);"S DE" ;P$(T)
6693 IFPA(T)=-999THENPA(T)=0
6694 PA(T)=PA(T)+X
6695 NEXT
6699 GOTO6592
6700 REM EPIDEMIE DE PESTE
6705 IFM(2)=1THENRETURN
6706 PRINTCHR$(147)
6710 M(2)=1
6712 S$="LA PESTE FRAPPE!+":GOSUB9100
6714 PRINT:GOSUB9200
6716 X=1
6718 FORT=1TO16
6720 IFTS(T,1)=2ANDTS(T,2)<>0ANDTS(T,2)<>-999THENX
=0:T=16
6722 NEXT
6724 Y=1
6726 IFDA(1)<>0ANDOA(1)<>-999THENY=0
6730 Z=(X+Y)*10)+5
6732 I$="VOUS AVEZ":IFX=0ANDY=0THEN6740
6734 IFX=1THENS$="ET PAS DE MEDECIN+":GOSUB9100:I$="A
ND"
6736 IFY=1THEN S$=I$+"PAS DE REMEDES+":GOSUB9100
6740 S$="BEAUCOUP DE MATELOTS SONT ATTEINTS":GOSUB9100
6745 PRINT:GOSUB9200
6750 X=0
6755 FORT=1TO16
6756 IFRND(1)<.3THEN6775
6760 IFTS(T,2)=0ORTS(T,2)=-999THEN6775
6765 TS(T,2)=TS(T,2)-Z
6770 IFTS(T,2)<1THENTS(T,2)=-999:X=X+1
6775 NEXT
6776 IFY=1THEN6780
6777 S$="1/2 DE VOS REMEDES ONT SERVI+":GOSUB9100
6778 DA(1)=INT((DA(1)/2)+.5)
6780 PRINT:GOSUB9200
6785 IFX=0THEN6797
6790 PRINT "ET";X;
6792 S$="MEMBRES D'EQUIPAGE MEURENT+"
6794 IFX=1THENS$="MEMBRE D'EQUIPAGE MEURT+"
6795 GOSUB9100
6796 PRINT:GOSUB9200
6797 S$=K$:GOSUB9100
6798 GETI$:IFI$=""THEN6798
6799 RETURN
```

Variantes de basic

Spectrum :
Procédez aux changements suivants :

```
6510 IFX=1 THEN GOSUB 6530
6511 IFX=2 THEN GOSUB 6700
6536 CLS
6578 INPUT I$ : LET I$ (1 TO 1)
6596 LET I$ = INKEY$ : IF I$ = "" THEN GOTO 6596
6706 CLS
6798 LET I$ = INKEY$ : IF I$ = "" THEN GOTO 6798
```

BBC Micro :
Procédez aux changements suivants :

```
6536 CLS
6596 I$=GET$
6706 CLS
6798 I$=GET$
```

Huile de coude

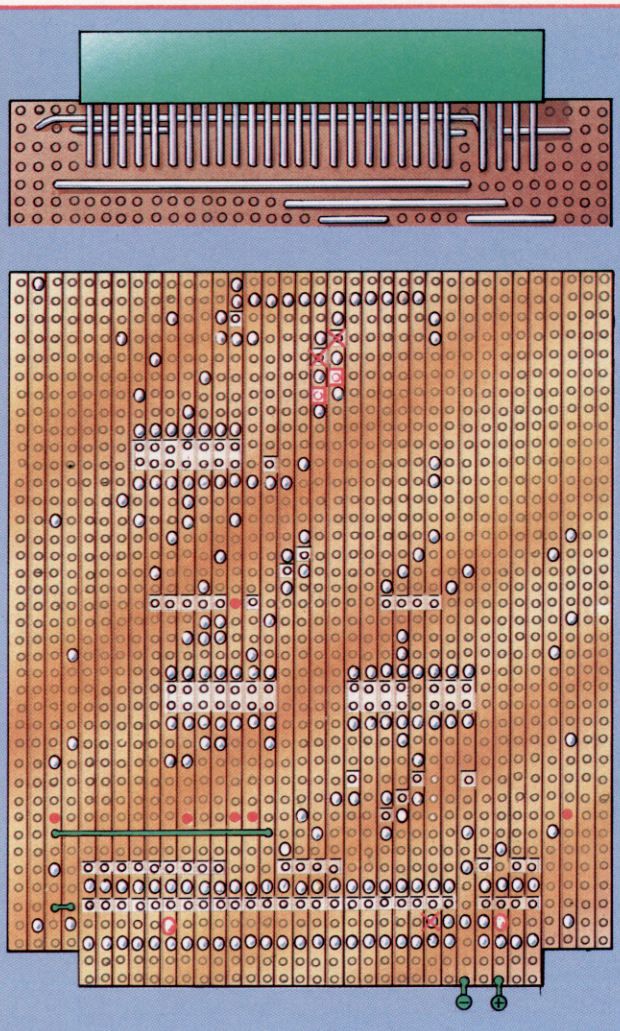
Les différences d'architecture entre les micros, impliquent des différences d'écriture des programmes de contrôle du bras de robot. Nous traitons ici du Commodore.

Leur séquence est de 1/60e de seconde. Le programme en code machine du Commodore intercepte le vecteur d'interruption et exécute sa propre routine, avant de donner le contrôle à la routine normale d'interruption. Ce programme est appelé « intercalaire ». La première partie du code machine concerne donc le remplacement du vecteur normal IRQ par le vecteur de la routine qui nous intéresse.

Après cela, le programme-contrôle prédéfini d'événement sera implémenté à chaque occurrence d'une interruption IRQ. Le code de contrôle de l'événement est pour sa part très semblable au programme de contrôle multiserveur. La routine en code machine reçoit l'information qui détermine la durée des impulsions moteur à partir d'une table à 8 octets, appelée ANGLE.

Du code supplémentaire sert à charger les positions pour ANGLE à partir d'une deuxième table, NOUVPOS, chargée à partir du programme de contrôle BASIC. Le programmateur de la séquence du bras utilise ce programme en code machine pour piloter le bras du robot. Il permet de diriger le bras à partir du clavier et de sauvegarder les positions clés, susceptibles d'être reprises par le bras ou stockées sur disque ou sur bande. Néanmoins, il faut tenir compte de deux éléments importants avec le Commodore 64 : d'abord, lors de la conduite du bras, les moteurs ont tendance à osciller. Cela est dû à l'interférence en provenance du composant vidéo VIC-II, avec l'émission normale d'interruptions. La manière la plus simple d'y remédier est de vider l'écran lors de l'utilisation du programme intercalaire.

Le deuxième problème vient de la lenteur du BASIC Commodore à lire la frappe au clavier (et à effectuer les connexions appropriées). Le sous-programme final en code machine est donc utilisé pour comparer les codes ASCII lus au clavier aux valeurs d'une table représentant les touches significatives. Lorsqu'une correspondance est établie, la valeur est transmise sous la forme d'un nombre pouvant être utilisé par l'instruction BASIC ON...GOSUB pour appeler la routine appropriée.



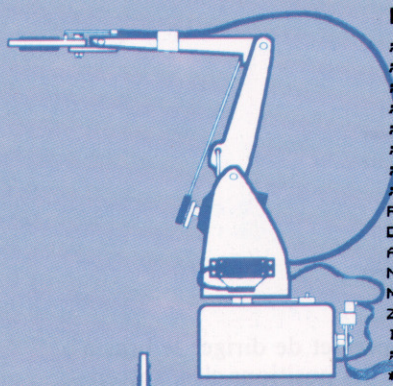
Erratum

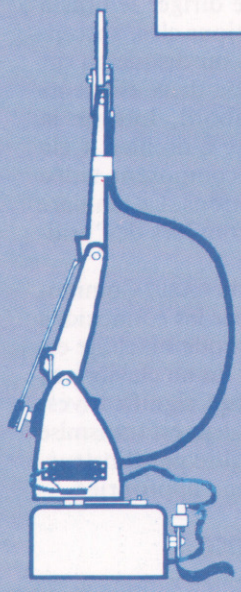
Le diagramme de disposition de la carte interface Spectrum donné précédemment comporte des erreurs. Voici les modifications à apporter : le long fil volant doit être reporté un trou plus à droite sur son extrémité droite, tout comme à l'extrémité gauche de la liaison adjacente. Le seul pontage de la deuxième rangée doit être déplacé d'une position sur la droite. L'extrémité droite de la courte liaison à la gauche de la deuxième rangée devrait être deux trous plus à droite. Notez également les modifications correspondantes sur le bord de cuivre de la carte.
(Cl. Kevin Jones.)

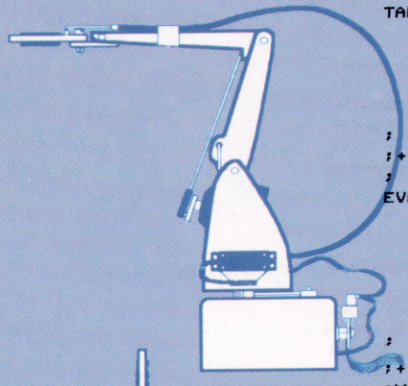
Programmeur de séquence de bras/Commodore 64

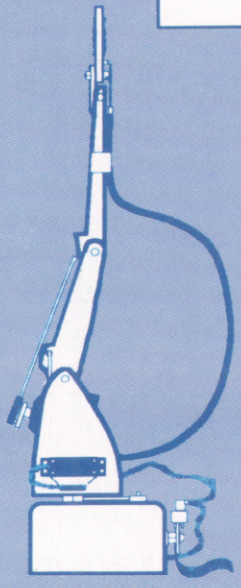
Chargeur basic

```
10 REM ********************************
20 REM **                            **
30 REM **   CHARGEUR BASIC POUR      **
40 REM ** LE CONTROLE DE BRAS CBM **
50 REM **                            **
60 REM ********************************
65 :
70 FOR I=49155 TO 49394
75 READ A:POKE I,A:CC=CC+A
80 NEXT I
90 READ CS:IF CS<>CC THEN PRINT "ERREUR DE PARITE":STOP
95 :
100 DATA169,255,141,3,221,169,62,141,1
110 DATA192,169,192,141,2,192,120,173
120 DATA20,3,174,1,192,141,1,192,142
130 DATA20,3,173,21,3,174,2,192,141,2
140 DATA192,142,21,3,169,16,133,251
150 DATA169,193,133,252,169,255,160,0
160 DATA145,251,136,208,251,88,96,8,72
170 DATA152,72,138,72,169,255,141,1
180 DATA221,162,7,169,255,24,106,72
190 DATA188,0,193,49,251,145,251,104
200 DATA202,16,243,160,48,136,208,253
210 DATA169,255,160,0,49,251,141,1,221
220 DATA200,208,248,162,7,169,255,188
230 DATA0,193,145,251,202,16,248,104
240 DATA170,104,168,104,40,108,1,192
250 DATA162,0,160,0,189,8,193,221,0
260 DATA193,240,14,176,6,222,0,193,76
270 DATA156,192,254,0,193,76,156,192
280 DATA200,232,224,4,208,228,32,169
290 DATA192,192,4,208,217,96,138,72
300 DATA152,72,160,255,174,0,192,136
310 DATA234,234,208,251,202,208,248
320 DATA104,168,104,170,96,0,255,255,0
330 DATA0,255,255,0,0,255,255,0,0,255
340 DATA255,0,0,72,138,72,152,72,32
350 DATA228,255,201,0,240,249,162,0
360 DATA221,191,192,240,7,232,224,17
370 DATA208,246,162,255,142,207,192
380 DATA104,168,104,170,104,96
390 DATA33097:REM CONTROLE DE PARITE*
```









Ian McKinnell

Listage d'assemblage

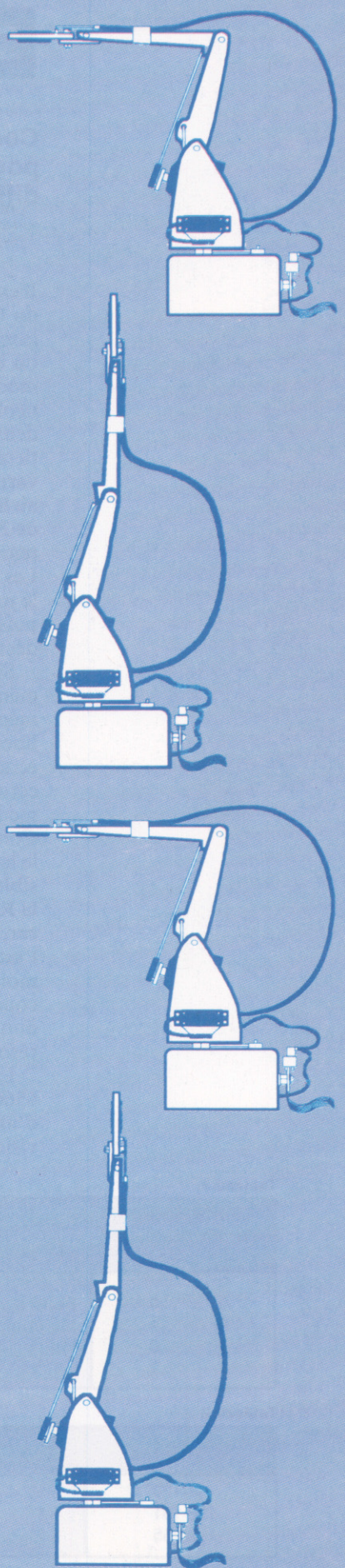
```
;++++++++++++++++++++++++++
;++++++++++++++++++++++++++
;++                      ++
;++ CBM ARM CONTROLLER   ++
;++                      ++
;++++++++++++++++++++++++++
;++++++++++++++++++++++++++
;
PORT = 56577 ;USER PORT DATA REGISTER
DDR=56579    ;USER PORT DATA DIRN REG
ANGLE=$C100  ;ANGLE VALUE LOCATION
NEWPOS=$C108 ;POSITION TABLE
MOTTAB=$C110 ;MOTOR LOOK UP TABLE
ZPAGE=$FB    ;0 PAGE POINTER TO TABLE
IRQVEC=$0314 ;IRQ VECTOR LOBYTE
;
*=$C000
DELFAC *=*+1 ;STORAGE FOR DELAY FACTOR
OURVEC *=*+2 ;STORAGE FOR OUR VECTOR
      LDA #$FF
      STA DDR      ;SET DDR TO OUTPUT
      LDA #<EVENT
      STA OURVEC   ;POINT TO EVENT
      LDA #>EVENT  ;HANDLER
      STA OURVEC+1
;
      SEI          ;INTERRUPTS OFF
      LDA IRQVEC   ;SWAP EXISTING
      LDX OURVEC   ;IRQ VECTOR FOR
      STA OURVEC   ;OUR VECTOR
      STX IRQVEC
      LDA IRQVEC+1
      LDX OURVEC+1
      STA OURVEC+1
      STX IRQVEC+1
;
;++++ INITIALISE TABLE ++++
;
      LDA #<MOTTAB
      STA ZPAGE
      LDA #>MOTTAB
      STA ZPAGE+1
;
      LDA #$FF
      LDY #$00
TABLE
      STA (ZPAGE),Y
      DEY
      BNE TABLE
      CLI          ;INTERRUPTS ON
      RTS
;
;++++ EVENT HANDLER ++++
;
EVENT
      PHP
      PHA          ;SAVE REGISTERS
      TYA          ;ON STACK
      PHA
      TXA
      PHA
;
;++ START PULSE, FOR SOME MOTORS IT
;MAY BE POSSIBLE TO START BEFORE FILLING
;TABLE AND SO REDUCE WAIT LOOP BELOW ++
;
      LDA #$FF
      STA PORT
;++ FILL TABLE WITH EXCEPTIONS ++
      LDX #$07
      LDA #$FF
      CLC
EXCEPT
      ROR A
      PHA          ;BIT PATTERN
      LDY ANGLE,X  ;GET MOTOR X OFFSET
      AND (ZPAGE),Y ;KEEP EXISTING PATTERN
      STA (ZPAGE),Y ;BUT MODIFIED FOR MOTOR X
      PLA
      DEX
      BPL EXCEPT
;++ TABLE IS NOW LOADED ++
      LDY #$30
WAIT
      DEY          ;FILL IN SOME
      BNE WAIT     ;TIME

      LDA #$FF     ;ALL PULSES ON
      LDY #$00
LOOP
      AND (ZPAGE),Y ;BUT MASK OFF WITH EACH
      STA PORT      ;TABLE ELEMENT IN TURN
      INY
      BNE LOOP
;
      LDX #$07
      LDA #$FF
CLEAR
      LDY ANGLE,X   ;CLEAR ALL EXCEPTIONS
      STA (ZPAGE),Y
      DEX
      BPL CLEAR
;++ ALL PULSES SHOULD NOW BE FINISHED ++
      PLA
      TAX
      PLA           ;RESTORE REGISTERS
      TAY
      PLA
      PLP
      JMP (OURVEC)  ;JUMP TO NORMAL
                    ;IRQ ROUTINE
;
;++++ SMOOTH SERVO MOVER ++++
;
LOOP2
     LDX #$00
     LDY #$00
LOOP1
     LDA NEWPOS,X ;NEW POSN FOR SERVO X
     CMP ANGLE,X  ;IS OLD POSN SAME?
     BEQ MOVED.   ;IF = THEN DO NOTHING
     BCS ADD      ;IF > THEN ADD
     DEC ANGLE,X  ;ELSE SUBTRACT
     JMP INCRX
ADD
     INC ANGLE,X
     JMP INCRX
MOVED
     INY
INCRX
     INX
     CPX #$04     ;ALL 4 SERVOS DONE
     BNE LOOP1    ;IF NOT NEXT SERVO
     JSR WAITER   ;SHORT PAUSE
     CPY #$04     ;IF Y=4 THEN ALL MOVED
     BNE LOOP2
     RTS
;
WAITER
     TXA
     PHA          ;SAVE X & Y REGS
     TYA
     PHA
     LDY #$FF
     LDX DELFAC   ;GET DELAY FACTOR
ENCORE
     DEY
     NOP          ;256*8 CLOCK CYCLES
     NOP          ;WAIT
     BNE ENCORE
     DEX          ;IF MORE DELAY THEN
     BNE ENCORE   ;REPEAT
     PLA
     TAY
     PLA
     TAX
     RTS
;
;++++ CHECK KEYBOARD ++++
;
KEYTAB *=*+16
RESULT *=*+1
GETIN=$FFE4
;
CHECK
     PHA
     TXA
     PHA
     TYA
     PHA
NOKEY
     JSR GETIN   ;GET A CHAR
     CMP #$00    ;NO CHAR?
     BEQ NOKEY
     LDX #$00
COMPAR
     CMP KEYTAB,X
     BEQ EXIT    ;IS CHAR IN TABLE?
     INX
     CPX #17
```

```
        BNE COMPAR
        LDX #$FF        ;SIGNAL NOT FOUND
EXIT
        STX RESULT      ;STORE KEYTAB OFFSET
        PLA
        TAY
        PLA
        TAX
        PLA
        RTS
```

Programmateur séquences bras

```
10 REM **********************
20 REM **********************
30 REM **                    **
40 REM ** PROGRAMMATEUR  **
50 REM ** SEQUENCES BRAS **
70 REM **                    **
80 REM **********************
90 REM **********************
95 :
96 ON=8:REM FOR CASS ON=I
97 IF A=0 THEN A=1:LOAD"BRARDB.HEX"ACTIVE 1
100 GOSUB 1000:REM ACTIVER
110 PRINT CL$
120 X=5:Y=6:GOSUB10000:PRINT VOUDRIEZ-VOUS:"
130 X=4:Y=10:GOSUB10000:PRINT"1....NOUVELLE SEQUENCE BRAS DE
PROGRAMME"
140 X=4:Y=12:GOSUB10000:PRINT"2....AJOUTER DEPLACEMENTS A UN
PROGRAMME"
150 X=4:Y=14:GOSUB10000:PRINT"3....EXECUTER A NOUVEAU UN
FICHIER"
160 X=4:Y=16:GOSUB10000:PRINT"4....QUITTER LE PROGRAMME"
170 GET G$:IF G$="" THEN 170
180 IF G$="1" THEN C=0:LM=0:REM RESTAURER TABLEAU PTRS
190 IF G$="1" OR G$="2é THEN GOSUB2000:GOSUB3000:G
OSUB4000:GOSUB5000
200 IF G$="3" THEN GOSUB6000:GOSUB4000
210 IF G$="4" THEN PRINTCL$:END
220 GOTO 110
500 REM **** FONCTIONS CLEFS ****
540 DX=DX+1:RETURN
550 DX=DX-1:IF DX<0 THEN DX=0
555 RETURN
560 P=PEEK(NP)+DX:IF P<256THEN POKENP,P
565 RETURN
570 P=PEEK(NP)-DX:IF P>0THEN POKENP,P
575 RETURN
580 P=PEEK(NP+1)+DX:IF P<256THEN POKENP+1,P
585 RETURN
590 P=PEEK(NP+1)-DX:IF P>0THEN POKENP+1,P
595 RETURN
600 P=PEEK(NP+2)-DX:IF P>0THEN POKENP+2,P
610 P=PEEK(NP+2)+DX:IF P<256THEN POKENP+2,P
615 RETURN
620 P=PEEK(NP+3)+3+DX:IF P<256THEN POKENP+3,P
625 RETURN
630 P=PEEK(NP+3)-3+DX:IF P<256THEN POKENP+3,P
635 RETURN
640 FOR I=0 TO 3:POKE NP+I,RX(C,I):NEXT I:RETURN
650 C=C+1:IF C>MC THEN C=0
655 RETURN
660 C=C-1:IF C<0 THEN C=MC
665 RETURN
670 FOR I=0 TO 3:RX(C,I)=PEEK(NP+I):NEXT:C=C+1:RET
URN
680 FOR I=0 TO 3:RX(C,1)=PEEK(NP+I):NEXT:C=C+1:RET
URN
999 :
1000 REM **** ACTIVER ****
1010 C=0:NS=8:REM NO. OF DEPLACEMENTS-1
1020 LM=0:DC=0:DF=10:REM FACTEUR DE RETARDEMENT
1030 MC=100:REM COMPTE MAX
1040 DIM RX(MC,NS):REM CPE TABLEAU POSITIONS
1050 CL$=CHR$(147):REM VIDER L'ECRAN
1060 DW$="":FOR I=1 TO 25:DW$=DW$+CHR$(17):NEXT I:
REM CURSEUR BAISSE
1065 POKE 650,128:REM ACTIVER LES TOUCHES POUR REPETITION
1070 OL=49152:NP=49416:REM RETARDER ET POSITIONNER LES
ADRESSES
1075 GS=49155:OFF=49170:REM INTERCALAIRE ON/OF/ ADRESSES SYST.
1077 MOVE=49281:REM DEPLACER LES ADRESSES SYSTEME DES SERVOS
1078 KEYTAB=49343:RESULT=49359:REM ADRESSES DES TOUCHES
1079 CHECK=49360:REM ADRESSE SYSTEME DE VERIFICATION DE TOUCHE
1080 REM ** LIRE DONNEES ASCII TOUCHES **
1090 FOR I=0 TO 14:READ A:POKE KEYTAB+I,A:NEXT I
1100 DATA 73,69,157,29,145,17,65,90,88,67
1110 DATA 82,78,66,83,81
1200 REM ** AUCUNE POSITION NOUVELLE **
1210 FOR I=0 TO 3:POKENP+I,0:NEXT I
1990 RETURN
1999 :
2000 REM **** INFORM ****
2010 PRINTCL$
2020 X=5:Y=5:GOSUB10000:PRINT"PLEASE USE :"
2030 X=1:Y=7:GOSUB10000:PRINT"TOUCHES-CURSEURS... POUR G/D ET
H/B 1ER BRAS
2040 X=1:Y=9:GOSUB10000:PRINT"A & Z.........FOR 2N
D ARM H/B"
2050 X=1:Y=11:GOSUB10000:PRINT"X & C.........FOR 2e BRAS
RAB"OUVERT/FERME"
2060 X=1:Y=13:GOSUB10000:PRINT"S.............POUR SAUVEGARDER
UNE POSITION
2070 X=1:Y=15:GOSUB10000:PRINT"Q.............POUR REVENIR AU
MENU
2080 X=1:Y=17:GOSUB10000:PRINT"R.............POUR ALLER SUR
UNE POSITION SAUVEGARDEE
2090 X=1:Y=19:GOSUB10000:PRINT"N & B.........SUIVANT ET COMPTE
ARRIERE
2100 X=1:Y=21:GOSUB10000:PRINT"E.............AFFECTER NOUVEAU
COMPTE
2110 X=1:Y=23:GOSUB10000:PRINT"I & D.........POUR INC. ET DEC.
LA VITESSE
2120 X=2:Y=2:GOSUB10000:PRINT"COUNT="+C:" ";
2130 FOR I=0 TO 3:PRINTRX(C,I);" ";:NEXT:PRINT
2135 GET A$:IF A$="" THEN 2135
2140 RETURN
2999 :
3000 REM **** PROGRAMME BRAS ****
3005 GOSUB7000:REM VIDER L'ECRAN ET COMMENCER PROG.
INTERCALAIRE
3010 POKE OL,1:REM METTRE FACTEUR DE RETARDEMENT A 1
3020 DX=8:REM MODIFIER CADENCE
3030 SYS CHECK:REM VERIFIER SAISIE
3035 R=PEEK(RESULT)+1IF R>15 THEN 3030:REM FRAPPE ILLEGALE
3037 ON R GOSUB 540,550,560,570,580,590,600,610,62
0,630,640,650,660,670
3190 IF C>LM THEN LM=C-1:REM RECORD COMPTE MAX JUSQU'ICI
3220 OC=C
3260 SYS MOVE:REM M/C DEPLACER LES SERVOS
3270 IF P=15 OR C>MC THEN GOSUB8000:RETURN:REM DESACTIVER
LE PROG. INTERCALAIRE ET TERMINER
3280 GOTO 3030:REM REPETE
3999 :
4000 REM **** REFAIRE UNE SEQUENCE ****
4010 PRINTCL$
4020 X=5:Y=23:GOSUB10000:PRINT REFAIRE UNE SEQUENCE O/N,
R REPETE"
4030 GET AN$:IF AN$<>"O" AND AN$<>"N" AND AN$<>"R"
THEN 4030
4040 IF AN$="N" THEN RETURN
4050 IF AN$<>"R" THEN X=5:Y=22:GOSUB10000:INPUT"FACTEUR DE
RETARDEMENT 1-255";DF
4060 IF<DF 0 OR DF>255 THEN 4050:REM VERIFIER ECHELLE
4070 POKE DL,DF:REM AFFECTER LE REGISTRE DU FACTEUR DE
RETARDEMENT
4075 GOSUB7000:REM VIDER L'ECRAN ET COMMENCEZ LE PROGRAMME
INTERCALAIRE
4080 FOR I=0 TO LM
4090 X=5:Y=2:GOSUB10000:PRINT"NO. IN SEQUENCE = ";
I;" ";
4100 FOR S=0 TO 3
4110 POKE NP+S,RX(I,S)
4120 NEXT S:SYS MOVE:REM M/C DEPLACER SERVOS
4130 NEXT I
4135 GOSUB8000:REM RESTAURER L'ECRAN ET DESACTIVER PROGRAMME
INTERCALAIRE
4140 GOTO 4010:REM REPETE
4999 :
5000 REM **** SAUVEGARDER UNE SEQUENCE ****
5010 PRINTCL$
5020 X=5:Y=10:GOSUB10000:PRINT SAUVEGARDER SUR FICHIER UNE
SEQUENCE O/N
5030 GET AN$:IF AN$<>"o" AND AN$<>"N" THEN 5030
5040 IF AN$="N" THEN RETURN
5050 X=5:Y=12:GOSUB10000:INPUT"NOMBRE FICHIER";FL$
5060 OPEN 3,DN,3,"00:"+FL$+",DATA,S,W":REM OPEN FI
LE
5070 PRINT#3,LM
5080 FOR A=0 TO LM:FOR B=0 TO 3
5090 PRINT#3,RX(A,B):NEXT B,A
5100 PRINT#3:CLOSE8
5120 RETURN
5999 :
6000 REM **** CHARGER UN FICHIER ****
6010 PRINTCL$
6020 X=5:Y=10:GOSUB10000:INPUT NOM DU FICHIER A CHARGER";F
L$
6040 OPEN 3,ON,3,"00:"+FL$+",DATA,S,R":REM OUVRIR FICHIER
6050 INPUT#3,C:LM=C
6060 FOR A=0 TO C:FOR B=0 TO 3
6070 INPUT#3,RX(A,B):NEXT B,A
6080 PRINT#3:CLOSE8
6100 RETURN
6999 :
7000 REM **** COMMENCER INTERCALAIRE ****
7010 POKE53265,PEEK(53265)AND 239:REM VIDER ECRAN
7015 SYS GS
7020 RETURN
7999 :
8000 REM **** FIN INTERCALAIRE ****
8010 POKE53265,PEEK(53265)OR 16:REM VIDER ECRAN
8015 SYS OFF
8020 RETURN
8999 :
10000 REM **** POSITIONNER CURSEUR EN X,Y ****
10010 PRINTCHR$(19);:PRINTTAB(X);LEFT$(DW$,Y);
10020 RETURN
```



Effets visuels

Considérons la puce contrôleur vidéo du Commodore 64 — VIC-II — pour voir comment on peut utiliser ses registres afin d'afficher différents modes et des effets visuels inhabituels.

Il existe huit modes graphiques de base sur l'écran du Commodore 64. En graphique basse résolution, le jeu de caractères peut se trouver en ROM ou en RAM, et être affiché dans l'un des trois modes possibles : le mode standard, le mode multicolore ou le mode couleur étendu. Il existe aussi deux modes haute résolution : standard et multicolore. De plus, il peut y avoir de nombreuses variantes : on peut régler l'écran pour 38 colonnes, au lieu des 40 habituelles, et/ou 24 lignes, au lieu de 25. Ce mode est normalement utilisé conjointement avec le défilement vertical ou horizontal. Les caractéristiques de défilement du C64 sont le mieux exécutées à partir du langage machine, puisque les données d'affichage doivent être décalées assez rapidement en RAM écran.

Si l'on utilise un écran haute résolution en même temps qu'un long programme, alors la mémoire peut faire prime. Le C64 autorise une liberté considérable dans l'emplacement des écrans haute et basse résolution, et nous commencerons donc par examiner comment le programmeur peut les déplacer tous deux en mémoire. Comme nous le verrons plus loin, si l'on utilise le langage machine pour le tracé, il est même possible de placer l'écran haute résolution *derrière* la ROM interpréteur BASIC. C'est très important, car, si l'on exécute seulement le langage machine, l'interpréteur ROM prendra 8 K d'espace mémoire valable qui ne servent à rien ! Par une coïncidence intéressante, ce sont exactement 8 K dont nous avons besoin pour un écran haute résolution.

La fonction de la puce d'interface vidéo 6566/67 (VIC-II) est de générer les données d'affichage vidéo, qui sont transmises à la télévision ou au moniteur. Pour faire cela, VIC-II doit être capable de lire des données de la RAM ou de la ROM. Il est bon de comprendre comment VIC-II obtient ses données, puisque cela détermine l'un des aspects les plus obscurs du Commodore 64.

Tableau 1

Configuration de bit de paire de pixels		Couleur	Déterminé par
0	0	Fond d'écran	53281 ($D021) Bits 0 à 4
0	1	Multicolore#1	53282 ($D022) Bits 0 à 4
1	0	Multicolore#2	53283 ($D023) Bits 0 à 4
1	1	Couleur d'avant-plan	Bits 0 à 3 du quartet couleur

Tableau 2

Code écran	Bit 7	Bit 6	Registre couleur de fond
64-127	0	1	D53282 ($D022) Étendu couleur 1
128-191	1	0	D53283 ($D023) Étendu couleur 2
192-255	1	1	D53284 ($024) Étendu couleur 3

Mode multicolore

Nous avons décrit précédemment le mode d'affichage basse résolution normal et la table d'implantation de bits. Ici, nous examinerons deux des autres façons dont on peut utiliser les graphiques machine.

En mode multicolore, il est possible d'avoir quatre couleurs dans une seule cellule de caractère, au lieu de deux. Mais le prix à payer est que la résolution horizontale est maintenant en paires de pixels au lieu de pixels uniques. Le mode multicolore peut être employé en graphique haute ou basse résolution, bien que les couleurs des points soient déterminées un peu différemment dans le mode haute résolution. Les POKEs suivants à partir du BASIC valident ou invalident le mode multicolore :

```
POKE 53270,PEEK(53270)OR16
POKE 53270,PEEK(53270)AND239
```

En mode basse résolution, quand le mode multicolore est mis, si le bit 4 du quartet (c'est-à-dire 4 bits ou un demi-octet) couleur associé est à 1, alors le caractère est interprété en mode multicolore, les 3 bits inférieurs déterminant la couleur. Cela signifie que ces caractères qui ont associé les quartets couleur dans l'intervalle 0 à 7 sont interprétés normalement ; si le code couleur se trouve entre 8 et 15, alors le caractère sera affiché en mode multicolore. Les couleurs des paires de pixels apparaissent dans le tableau 1.

En modifiant les contenus des adresses 53282 et 53283, nous pouvons instantanément changer la couleur de toutes les paires de pixels multicolores associées. Il faut noter que le mode multicolore marche mieux avec les caractères définis par l'utilisateur — c'est-à-dire que les configurations de bits prennent en compte le fait qu'ils sont interprétés comme des paires.

Un autre mode graphique disponible pour le programmeur du C64 est le mode couleur étendu. Il permet de contrôler la couleur de fond des 64 premiers caractères dans la matrice caractère. Le mode couleur étendu ne peut être utilisé en conjonction avec le mode multicolore. Les lignes BASIC suivantes valident et invalident respectivement ce mode :

```
POKE 53265,PEEK(53265)OR64
POKE 53265,PEEEK(53265)AND191
```

Le même caractère peut apparaître à l'écran avec jusqu'à quatre différentes couleurs de fond, dont l'une est la couleur du fond d'écran. Les caractères restants ne peuvent être affichés en mode couleur étendu, puisque les bits 6 et 7 du code écran servent à contrôler indirectement la couleur du caractère. Par exemple, le code écran pour « A » est 1, et celui d'un champ inverse « A » est 65. Cependant, si nous faisions POKE 65 dans l'écran en mode couleur étendu, nous ne verrions pas un champ inverse « A », mais un « A » normal avec une couleur de fond déterminée par le contenu de l'adresse 53282 ($D022). De même, en faisant POKE 129 dans l'écran, on produirait un « A » normal, mais avec une couleur de fond déterminée par le contenu de l'adresse 53283 ($D023). Le tableau 2 montre comment codes écran et registres couleur sont reliés.

Organisation mémoire

La puce VIC-II voit une carte de la mémoire différente du 6510, et bien plus simple. A tout instant, VIC-II ne peut voir que quatre blocs de 16 K de mémoire. Nous pouvons assimiler un bloc de 16 K à la « fenêtre » de VIC-II sur la mémoire. L'adresse de base de cette fenêtre peut prendre l'une des quatre valeurs sous contrôle logiciel.

```
1000 REM**SÉLECTION BLOC POUR FENÊTRE VIC**
1010 POKE 56578,PEEK(56578)OR3:REM DDR CIA#2 BITS 0,1 EN
     SORTIE
1020 WD=3:REM SÉLECTIONNE FENÊTRE NORMALE
1030 POKE 56576,(PEEK(56576)AND252) OR WD:REM PORT A
     CIA#1 BITS 0,1
```

Ici, WD (pour « WinDow » : fenêtre) peut prendre des valeurs comprises entre 0 et 3. A tout moment, la valeur de WD en cours est trouvée par PEEK(56576)AND3. L'adresse correspondante du bas de la fenêtre 16 K en mémoire est calculée à l'aide de la formule : WB=16384*(3-WD), où la valeur de WD donne les adresses correspondantes :

WD	Adresse de début de fenêtre
0	49152 ($C000)
1	32768 ($8000)
2	16384 ($4000)
3	0 ($0000)

A l'intérieur de la fenêtre VIC-II, le 6510 s'attend à voir la mémoire écran et une image ROM de caractère en basse résolution (ou une donnée haute résolution si c'est le mode HR qui est sélectionné). Il peut aussi avoir besoin de trouver des données de lutins et, le cas échéant, de voir à l'intérieur de la fenêtre. Les pointeurs pour chacun des huit lutins sont placés à la fin de la mémoire écran (et doivent être déplacés avec l'écran).

Le décalé du début de la mémoire écran à partir de la base de la fenêtre VIC-II en cours est contrôlé par les quatre bits supérieurs du registre de contrôle VIC-II à l'adresse 53272 ($D018). En utilisant ces quatre bits, nous pouvons placer l'écran dans n'importe lequel des blocs de 16 K à l'intérieur de la fenêtre.

```
1040 REM**SÉLECTION DECALÉ ÉCRAN A PARTIR DE BASE
     DE FENÊTRE**
1050 SO=1:REM CELA SÉLECTIONNE DECALÉ NORMAL
1060 POKE 53272,(PEEK(53272)AND15)OR 16*SO
```

Ici, SO peut prendre des valeurs comprises entre 0 et 15. A tout moment, la valeur de SO en cours est trouvée à l'aide de PEEK(53272)AND15. L'adresse correspondante de la base de l'écran en cours dans la mémoire est calculée par l'expression SC=WB+1024+SO (base de fenêtre plus décalée), ou :

```
SC=16384*(3-(PEEK(56576)AND3)
+64*(PEEK(53272)AND240)
```

Le problème est que, en déplaçant l'écran, la RAM couleur ne se déplace pas. Donc, si nous voulons avoir un autre écran, il nous faut une petite routine en langage machine pour échanger la mémoire couleur avec un tampon de manière appropriée.

Un autre facteur à considérer est que, si vous voulez faire PRINT sur le nouvel écran, il est alors nécessaire de le dire au système d'exploitation (au lieu de la puce vidéo) où se trouve le nouvel écran. Cela peut être fait par POKE (ou STA) dans l'adresse 648 ($0288), qui est un pointeur devant soutenir l'octet hi de l'adresse de base de la mémoire écran. Si SC est calculé comme ci-dessus, le code BASIC suivant fera l'affaire :

```
POKE 648,INT(SC/256)
```

Pour sélectionner l'adresse de base de la matrice caractère ou de l'écran HR, nous utilisons le code suivant :

```
1070 REM**SÉLECTION HR/MC DÉCALÉ DE BASE DE
     FENÊTRE**
1080 HO=4:REM CELA SÉLECTIONNE DÉCALÉ NORMAL
1090 POKE 53272,(PEEK(53272)AND240)OR 2*HO
```

En principe, HO peut prendre toute valeur comprise entre 0 et 7, mais en pratique d'autres facteurs limitent les options. Avec l'équivalent WD à 1 ou 3, nous ne pouvons permettre à HO de prendre la valeur 2 ou 3, puisque c'est là que VIC-II voit l'image de caractère en ROM normale. Par ailleurs, de grandes valeurs de HO mettront le haut de la mémoire HR hors d'atteinte de VIC-II. L'adresse correspondante de la base de l'écran matrice de caractère/haute résolution (MC/HR) en mémoire est calculée comme suit :

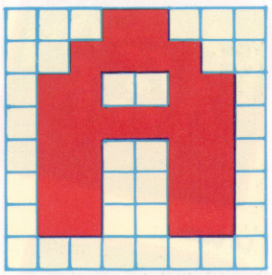
CM=WD+2048*HO (base de fenêtre plus décalé).

ou :

```
CM=16384*(3-(PEEK(56576)AND3))+1024
  *(PEEK(53272)AND14)
```

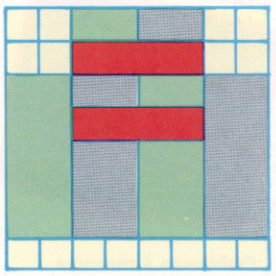
En mettant ces registres, nous pouvons déplacer toute la mémoire d'affichage vidéo de manière appropriée à notre programme.

Coordonnée couleurs
En mode d'affichage normal du C64, chaque bit mis à 1 dans les 8 octets qui définissent un caractère est affiché dans la couleur de fond en cours. Tous les bits mis à zéro sont affichés dans la couleur écran de fond. Lorsque le mode multicolore est sélectionné, les configurations de bits ne sont plus interprétées comme bits individuels mais comme paires. (Cl. Liz Dixon.)



« A » en mode d'affichage normal

0	0	0	1	1	0	0	0
0	0	1	1	1	1	0	0
0	1	1	0	0	1	1	0
0	1	1	1	1	1	1	0
0	1	1	0	0	1	1	0
0	1	1	0	0	1	1	0
0	1	1	0	0	1	1	0
0	0	0	0	0	0	0	0



« A » en mode multicolore

00	01	10	00
00	11	11	00
01	10	01	10
01	11	11	10
01	10	01	10
01	10	01	10
01	10	01	10
00	00	00	00

Clé

01 MULTICOLORE 1

10 MULTICOLORE 2

11 COULEUR D'AVANT-PLAN

00 COULEUR D'ARRIÈRE-PLAN

Un conte de fée

Après avoir créé des jeux très populaires, Ultimate vient d'introduire un type de jeu entièrement différent, Knight Lore, avec des graphiques tridimensionnels.

La chambre du sorcier

La botte

La tasse

Problèmes à résoudre
L'objectif de Knight Lore est d'apporter jusqu'au chaudron les objets nécessaires, afin d'empêcher le chevalier d'être transformé en loup-garou. Pour ce faire, le chevalier doit d'abord rendre visite au sorcier pour déterminer ce dont il a besoin. La recherche peut alors commencer. Bien que les objets puissent changer de position, ils se trouvent toujours dans la même pièce.
(Cl. Ian McKinnell.)

Ultimate a toujours eu la réputation de produire des jeux de qualité pour le Sinclair Spectrum. Cette société a créé une série complète de jeux-labyrinthes style « arcade » dans lesquels le joueur doit se frayer un chemin à travers de nombreuses pièces habitées par des créatures étranges et mystérieuses. Pendant ce temps, le joueur doit également découvrir divers objets, talismans et bibelots mystérieux, qui, une fois rassemblés, mettront fin au jeu. Atic Atac et Sabre Wulf en sont des exemples.

Malgré la popularité de ces jeux auprès des utilisateurs de Spectrum, la société a maintenant décidé de s'éloigner des jeux d'aventures bidimensionnels comme Sabre Wulf. Bien que semblable aux créations antérieures d'Ultimate, Knight Lore est très différent. Le scénario est le suivant : un chevalier (en fait, le héros de plusieurs aventures précédentes, Sabre Man) doit trouver le château d'un sorcier afin d'obtenir la formule magique qui l'empêchera d'être transformé en loup-garou. Le joueur dispose de quarante jours et de quarante nuits pour trouver la solution; le déroulement de ceux-ci est affiché au bas de l'écran; et dans le coin inférieur droit apparaît une fenêtre renfermant un « cadran solaire ». Pendant le jour, le joueur est Sabre Man. Cependant, lorsque la lune se lève dans la fenêtre, il est transformé de façon assez spectaculaire en loup-garou. Heureusement, cette transformation ne gêne en rien les déplacements du joueur, sauf au cours de la transformation elle-même, qui peut poser un problème si vous êtes à ce moment précis poursuivi par une gargouille.

Au début du jeu, vous remarquerez certainement qu'il est tridimensionnel. Ces graphiques sont les meilleurs qui aient été produits sur le Spectrum, même en incluant Ant Attack. Cependant, vous ne pouvez changer l'angle de vision sur Knight Lore; mais cette contrainte est compensée par la construction ingénieuse des diverses pièces.

Knight Lore n'est pas un jeu d'aventures au sens propre, mais une série d'énigmes logiques. De ce point de vue, Knight Lore ressemble à un véritable jeu d'aventures, posant des problèmes complexes qui doivent être résolus avant de poursuivre plus avant. Bien que de nombreuses pièces soient vides, d'autres vous placent devant diverses difficultés avant que vous n'atteigniez la sortie.

Les énigmes présentent divers niveaux de difficultés. Certaines sont relativement faciles, ou réellement complexes, et d'autres sont presque insolubles. Vous pouvez entrer dans une pièce contenant des blocs sur lesquels ont été placées de grosses sphères hérissées de clous. Si vous tentez de traverser les blocs, votre enthousiasme risque d'être refroidi assez rapidement — la solution consiste simplement à contourner les blocs! Une autre astuce utilisée par les programmeurs consiste à employer une perspective déformée que l'on obtient lorsque l'on voit une représentation bidimensionnelle d'un espace tridimensionnel. Souvent, les blocs, qui semblent posés sur le sol, flottent en fait librement dans l'air et, si Sabre Man heurte un de ces blocs, il perd automatiquement une vie et doit de nouveau essayer de traverser la pièce.

Dans de nombreuses pièces, des objets doivent être recueillis, comme des pierres précieuses, des coupes et des potions. Pour les ramasser lorsque vous les découvrez, vous n'avez qu'à bondir sur l'objet et ramener le manche à balai vers l'arrière. Cela place l'objet dans votre « inventaire », qui est affiché dans le coin inférieur gauche de l'écran. Cette manœuvre peut vous permettre d'obtenir une hauteur supplémentaire lors de l'escalade d'un mur trop haut.

Sabre Man peut se déplacer dans quatre directions en utilisant soit le clavier, soit le manche à balai. Les sauts sont effectués par la pression d'une des touches de la troisième rangée du clavier ou du bouton de mise à feu. La correction de l'orientation de Sabre Man est essentielle pour exécuter un jeu avec succès — un saut effectué dans la mauvaise direction par exemple pourrait être fatal. Il se peut que vous deviez consacrer les premiers jeux à vous familiariser avec la commande de l'orientation puisque le manche à balai est très sensible et réagit à la moindre pression.

Knight Lore est un jeu fascinant, d'aussi bonne qualité que les jeux Ultimate précédents. Même les joueurs qui n'aiment pas résoudre des énigmes seront fascinés par le jeu, puisque les solutions de nombreux problèmes n'exigent qu'une réaction rapide au clavier ou sur le bouton de mise à feu, tout en demandant une attention et une vivacité permanentes.

Knight Lore : Pour le Spectrum 48 K.
Éditeurs : Ashby Computers and Graphics Ltd.
Auteurs : Ultimate Play the Game.
Manches à balai : optionnel.
Format : cassette.

Page manquante
(publicité)

Page manquante
(publicité)